WORCESTER POLYTECHNIC INSTITUTE

DOCTORAL DISSERTATION

Robot Autonomy for Scrap Cutting in Metal Recycling

Author:
James AKL

Advisor:
Prof. Berk CALLI

A dissertation submitted to the
Robotics Engineering Department

of the
Worcester Polytechnic Institute

in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Robotics Engineering

APPROVED:

Prof. Nitin J. SANKET, Committee ______________________

Robotics Engineering, Computer Science, Electrical & Computer Engineering *WPI*

Prof. Yunus D. TELLIEL, Committee ______________________

Anthropology & Rhetoric, Humanities & Arts *WPI*

Prof. William R. MICHALSON, Committee ______________________

Robotics Engineering, Computer Science *WPI*

Prof. Berk CALLI, Advisor ______________________

Robotics Engineering, Computer Science *WPI*

Declaration of Authorship

I, James AKL, declare that this dissertation titled, "Robot Autonomy for Scrap Cutting in Metal Recycling" and the work presented therein are my own.

I confirm that:

- This work was done wholly while in candidature for a Doctor of Philosophy in Robotics Engineering, at Worcester Polytechnic Institute.
- This research is the result of my work at the Manipulation & Environmental Robotics Lab, under the supervision of Prof. Berk CALLI.
- Where I have consulted, utilized, or adapted the published work of others, this is clearly attributed and the source is cited.

Signed: **James AKL**

Date: **July 26, 2023**

"Those of us concerned with developing new technology should consider ourselves to have a major undertaking to try to meet the expanding needs of the increasing number of people in the world with its finite resources and environments constraints."

— Harold H. CHESTNUT, 1981.

"As always, machine's triumph was a human triumph, something we tend to forget when humans are surpassed by our own creations."

— Garry K. KASPAROV, 2017.

"This isn't about replacing human thinking with machine thinking. Rather, in the era of cognitive systems, humans and machines will collaborate to produce better results, each bringing their own superior skills to the partnership."

— John E. KELLY III, 2013.

WORCESTER POLYTECHNIC INSTITUTE

Abstract

School of Engineering

Robotics Engineering Department

Doctor of Philosophy in Robotics Engineering

Robot Autonomy for Scrap Cutting in Metal Recycling

by James AKL

This dissertation develops an automation framework to address the challenges, problems, and opportunities of the metal recycling industry. This is achieved by integrating a variety of components and functionalities into a diverse cognitive architecture. The aim is to endow robotic systems with task-specific autonomy against four main problems found in metal scrap cutting and recycling. These are: (1) Cutting path generation, using viewpoint planning and active perception; (2) Autonomous oxy-fuel cutting, using visual feedback for conditioning and control; (3) Cutting task validation, using learning-based inference via neural network models; and (4) Safe structural disassembly, using sequential decision planning. In this dissertation, we formalize and discuss the design and evaluation of each of these functionalities. Additionally, we analyze the broader impacts of this research from a socio-technical, economic, and ethical perspective. Ultimately, while this proposed framework is tailored towards metal recycling, many of the components' underlying techniques may be applicable or transferable to tasks of similar nature. In effect, we demonstrate the leverage and flexibility of diverse component-based architectures for augmenting an agent's capability, intelligence, and autonomy towards a particular goal.

Acknowledgements

To my Doctoral Advisor, thank you Berk, for all the wisdom and the lessons, and for bearing and wrestling through, all while remaining cheerful and true to yourself. During these years, your unyielding individuality has embodied the attitude of being fun while focused, carefree while competitive, and dynamic while dependable.

To my Dissertation Committee, Prof. Telliel, Prof. Sanket, and Prof. Michalson, thank you for your invaluable patience, feedback, and support throughout this research. Your guidance has been crucial in shaping the maturity and rigor of this work.

To our wonderful faculty at WPI's Robotics Engineering Department, thank you for fostering such a nurturing environment for learning, research, teaching, and growth. I express my thanks to Prof. Jing Xiao for having facilitated much of my progress.

To our dependable admins at WPI's Robotics Engineering Department, Pauline, Jessica, and Sharon, and formerly Kristen and Kat, you have helped push us all forward, all while fueling us with your warmth and positivity.

To WPI's lab managers and safety specialists, Ian Anderson, Christopher Stoney, and formerly Timothy Reilly, you have facilitated the more difficult logistics of this research and have inspired us by upholding the highest safety standards.

To WPI's International House, your endless support has enabled the success of many. You helped make WPI a home for everyone, and for that I am thankful. I extend my special thanks to Colleen Callahan-Panday for her support in finalizing my program.

To WPI's Department of Graduate Studies, thank you for your support and for enabling me to finish this work. In particular, thank you Rory Flinn and Terri Camesano for all your pivotal help and guidance in finalizing my program.

To the WPI community, you have exemplified the power of cooperation, communal efforts, and acceptance. You have helped me learn and grow as an individual.

To EMR Group, thank you Roger Morton, Andrew Sheppard, Wayne Kelsall, Chris Green, Troy Berry, Alex Mata, Alejandra Zamora, and all the excellent workers at Brownsville, TX. Without you all, this research would not have been possible.

To the FORW-RD NRT and I/UCRC ROSE-HUB programs at WPI, with special thanks to Kristen Keane and Prof. Cagdas Onal, for fostering my academic and professional development with a multitude of learning opportunities.

To my peers at the Manipulation & Environmental Robotics Lab, I extend my thanks:

- To all the PhD students I have shared these years with: Abhinav Gandhi, Albert Enyedy, Fadi Alladkani, Galen Brown, and Sreejani Chatterjee. I will fondly remember our time together.
- To all the MS students I have collaborated with and supervised during "Directed Research": Akshay Laddha, Arvind Sureshkumar, Chinmay Todankar, Pratyush Sahoo, Shreedhar Kodate, Sumanth Pericherla, and Yash Patil.
- To all the undergraduate students and high-school interns I have worked with and mentored in research: Allen Hatchett, Alyssa Moore, and Maya Angeles; as well as, Aditya Gupta and Ansh Motiani.

To my ally, Vladimir Vakhter, for your company and camaraderie during these years. You have made the struggles feel ever so endearing and memorable.

To my mother, Micha, for her indomitable support, her motivating trust, and her endless love. Through your decades of fostering my growth, nurturing my ambition, and sharpening my focus, I bestow this success upon you as its enabler.

To my elder, Georges, you are unable to witness these days, but I give you my deepest thanks. Onwards and upwards, I will parlay the decades of your efforts.

To my sisters, Joyce and Jennifer, for their cheers and their encouragement. Despite the distances and the years, you have preserved our familial atmosphere. Your support has been a consistent tailwind throughout this journey.

To my sweetheart, Aya, for your undying patience, tender warmth, and precious love. You have fueled my endurance, my strength, and my humanity with your loving trust and have grown my spirit in all matters transformative and purposeful.

To my landlords, Martin and Carol Sigel, for your trust, your care, and your kindness. You have provided me with much stability during times of tumultuous change.

I extend my gratitude to all those who enrich the wider human condition. From accessible learning materials, to open-source tooling, to Good Samaritans, you give me hope towards a better Humanity.

Contents

List of Figures

List of Tables

List of Abbreviations

Adam	**Ada**ptive **m**oment estimation
BIM	**B**uilding **I**nformation **M**odeling
CAD	**C**omputer **A**ided **D**esign
CBRN	**C**hemical, **B**iological, **R**adiological, and **N**uclear
CCD	**C**harge-**C**oupled **D**evice
C-NBV	**C**onstrained **N**ext **B**est **V**iew
CNC	**C**omputer **N**umerical **C**ontrol
CNN	**C**onvolutional **N**eural **N**etwork
COVID-19	**CO**rona**VI**rus **D**isease 20**19**
CP	**C**ombustion **P**ool (class label)
CPU	**C**entral **P**rocessing **U**nit
DOF	**D**egree-**O**f-**F**reedom
EPA	**E**nvironmental **P**rotection **A**gency
E-NVP	**E**xtrapolated **N**ext **V**iew **P**lanning
ES	**E**xhaustive **S**earch
GMAW	**G**as **M**etal **A**rc **W**elding
G-NBV	**G**uided **N**ext **B**est **V**iew
GPU	**G**raphics **P**rocessing **U**nit
GS	**G**reedy **S**earch
GTAW	**G**as **T**ungsten **A**rc **W**elding
LED	**L**ight-**E**mitting **D**iode
MAG	**M**etal **A**ctive **G**as
NA	**N**ot **A**pplicable (class label)
NBV	**N**ext **B**est **V**iew
NURBS	**N**on-**U**niform **R**ational **B**-**S**plines
NVP	**N**ext **V**iew **P**lanning
OCTS	**O**xy-fuel **C**utting **T**ask **S**tate
OMPL	'**O**pen **M**otion **P**lanning **L**ibrary'
OSHA	**O**ccupational **S**afety and **H**ealth **A**dministration
PP	**P**reheating **P**ool (class label)
PPE	**P**ersonal **P**rotective **E**quipment
R&D	**R**esearch and **D**evelopment
RGB	**R**ed-**G**reen-**B**lue
RGB-D	**R**ed-**G**reen-**B**lue and **D**epth
ROS	'**R**obot **O**perating **S**ystem'
RRT	**R**apidly-exploring **R**andom **T**ree
RS	**R**andom **S**earch
SDP	**S**equential **D**ecision **P**roblem
SiLU	**Si**gmoid **L**inear **U**nit
TF	**T**orch **F**lame (class label)
TIG	**T**ungsten **Inert**-**G**as
WAAM	**W**ire-**A**rc **A**dditive **M**anufacturing

List of Symbols

Chapter 4

${}^{\mathsf{b}}C_k$	Point cloud acquired at step k in the base frame b
${}^{\mathsf{c}}C_k$	Point cloud acquired at step k in the camera frame c
${}^{\mathsf{b}}C_{\mathsf{total},k}$	Cumulative point cloud at step k in the base frame b
$\mathrm{coverage}_G({}^{\mathsf{b}}C)$	Coverage of a cloud ${}^{\mathsf{b}}C$ over a ground truth cloud ${}^{\mathsf{b}}G$
${}^{\mathsf{b}}D_{\mathsf{total},k}$	Feature point cloud at step k in the base frame b
${}^{\mathsf{b}}\mathbf{d}_k$	Position of the viewpoint at step k
d_{total}	Total displacement between visited viewpoints
$\mathcal{F}_k$	Search frontier at step k
$\hat{f}_k$	Centroid of the nearest frontier cluster
$\mathcal{G}_k$	Voxel grid generated at step k
$\mathrm{H}(o_g \mid s_k)$	Information entropy of the occupancy o_g given the viewpoint s_k
$h(g \mid s_k)$	Information entropy decrease at the cell g given the viewpoint s_k
k	Current exploration step
k_{stop}	Termination step
$n_{\mathsf{extremities}}$	Number of drawing extremities encountered
$n_{\mathsf{viewpoints}}$	Total number of visited viewpoints or planning steps
o_g	Binary occupancy random variable at the cell g
${}^{\mathsf{b}}\mathbf{p}$	Unordered point in the base frame b
$p_v(g \mid s_k)$	Visibility probability of the cell g from the viewpoint s_k
$p_\phi(g)$	Feature probability of the cell g
$Q(\cdot)$	Viewpoint quality metric
${}^{\mathsf{b}}\mathbf{q}_{\mathsf{xyz}}(t)$	Ordered point in the base frame b indexed by t
$\mathcal{R}_k$	Cells traversed by a ray cast at step k
${}^{\mathsf{b}}\mathrm{R}_k$	Orientation of the viewpoint at step k
$\mathcal{S}_k$	Search space at step k
$S(f)$	Search space primitive at frontier point f
${}^{\mathsf{b}}s_k$	Candidate viewpoints at step k in the base frame b
${}^{\mathsf{b}}\mathrm{T}_k$	Camera pose at step k in the base frame b
Δt	Total planning duration
$\overline{\Delta t}$	Average step planning duration
v_{g,s_k}	Binary visibility random variable of the cell g from the viewpoint s_k
$\mathrm{vox}(\cdot)$	Point cloud voxelization function
α_ϕ	Exponential decay parameter
$\delta_{\mathsf{distance}}$	Extrapolation distance (offset)
δ_Q	Viewpoint quality threshold
δ_{size}	Point cloud size difference threshold
λ	Convex combination parameter for the viewpoint gain and cost
ϕ_g	Feature indicator random variable at the cell g
$\phi_\xi(\cdot)$	Point cloud filtering function with parameter ξ

Chapter 5

$\mathcal{C}_{\mathbf{M}}$	Set of contours in the processed image $\mathbf{M}$
$C_{\mathsf{cal},k}$	Predicate k for the vision system calibration task
$C_{\mathsf{con},k}$	Predicate k for the metal surface conditioning task
c	Heat pool convexity
c^*	Reference value for heat pool convexity
c_p	Color value at the pixel p
$\operatorname{conv} K$	Convex hull of the contour K in the processed image
e_s	Combustion state error
$g(p)$	2D Gaussian filter applied at a pixel p in the processed image
I	Heat pool absolute intensity
$\tilde{I}$	Heat pool scaled intensity
I_{cal}	Heat pool baseline intensity
$\tilde{I}_{\mathsf{sat}}$	Heat pool saturation intensity
i	Heat pool normalized intensity
i^*	Reference value for heat pool intensity
K	Contour of the heat pool in the processed image (blue layer)
$\lvert K \rvert$	Area of the contour K in the processed image
k	Strictly positive control gain
$\mathbf{M}$	Processed image in array form
$\mathbf{M}_k$	Channel $k \in \{\mathsf{R}, \mathsf{G}, \mathsf{B}\}$ of the processed image $\mathbf{M}$
$\mathcal{N}$	Set of nonzero pixels in the processed image $\mathbf{M}$
$\mathcal{N}_k$	Set of nonzero pixels in channel $k \in \{\mathsf{R}, \mathsf{G}, \mathsf{B}\}$ of the processed image $\mathbf{M}$
N_{cal}	Number of nonzero pixels in the processed image during calibration
p	Pixel in the processed image
s	Heat pool combustion state
s^*	Desired heat pool combustion state
T_{cal}	Time window for the vision system calibration task
T_{con}	Time window for the metal surface conditioning task
t	Time (for the dynamical systems)
V	Lyapunov candidate function
v	Tangential velocity of the heat source on the reference path
v^*	Desired tangential velocity
v_{max}	Maximum velocity constraint
$\dot{v}_{\mathsf{max}}$	Maximum acceleration constraint
$w(p)$	Color-weighing function applied at a pixel p
(x_c, y_c)	Pixel coordinates of the torch flame centroid
(x_p, y_p)	Pixel coordinates of the pixel p
δ_i	Threshold value for the ‘low intensity gradient’ predicate
θ_c	Threshold value for the ‘high convexity’ predicate
θ_k	Threshold value for the binary thresholding in channel $k \in \{\mathsf{R}, \mathsf{G}, \mathsf{B}\}$
θ_N	Threshold value for the ‘heat pool size’ predicate
$\theta_{\mathsf{RL}}, \theta_{\mathsf{RU}}$	Threshold values for the ‘red color proportion’ predicate
λ	Convex combination parameter for heat pool features
σ_X, σ_Y	Gaussian decay factors in each image coordinate
ϕ	Map from the velocity v to the combustion state s

Chapter 6

$\mathtt{B}_i$	Functional block i in the model
$\mathtt{Conv}_{ij}$	Convolutional layer j of block i
CP	Combustion pool (class label)
$\mathtt{Dense}_{ij}$	Dense (fully-connected) layer j of block i
$\mathtt{Drop}_{ij}$	Dropout layer j of block i
$f(\cdot)$	Convolutional neural network model
L	Categorical cross-entropy loss function
N	Total number of training samples
NA	Not applicable (class label)
$\mathtt{Pool}_{ij}$	Max-pooling layer j of block i
PP	Preheating pool (class label)
TF	Torch flame (class label)
Vec	Vectorization function
$\mathbf{X}$	Image input to the classifier
$\mathbf{X}_c$	The c-channel of the input image $\mathbf{X}$
$\hat{\mathbf{y}}$	Class scores from the classifier
$\mathbf{y}_k$	True label of the k^{th} training input
α_c	One-sided threshold for channel c
$\boldsymbol{\theta}$	Mapping parameters (model weights)
σ	Sigmoidal Linear Unit (SiLU) activation function

Chapter 7

$\mathcal{A}$	Set of all decision sequences
$\mathcal{A}_k$	Set of available decisions remaining at step k
$\mathbf{a}$	Decision sequence
$\mathbf{a}^*$	Optimal (safest) decision sequence
$\mathbf{a}^\dagger$	Greedy solution (decision sequence)
$\mathbf{a}_{\mathsf{wc}}$	Worst-case solution (decision sequence)
a_k	Decision at step k
$\mathbf{C}$	Sequence of the cutter's prescribed positions
$\mathsf{C}(n)$	Total number of decision evaluations for an input of size n
$\mathbf{c}_k$	Cutter's prescribed position at decision step k
$\mathbb{E}[S(\mathbf{a})]$	Expected safety of the random solution (decision sequence)
$\mathcal{F}$	Free space
f	Environment dynamics (physics-based simulation)
$\mathcal{I}$	Immobile space
I_3	Identity matrix of dimension 3×3
k	Decision step
$\mathcal{L}$	Cutter's position locus
m	Cutting margin
$\mathcal{N}(\mathbf{c}_k, r^3\mathrm{I}_3)$	Trivariate Gaussian distribution with mean $\mathbf{c}_k$ and covariance $r^3\mathrm{I}_3$
n	Total number of cutting locations in the partitioned input structure
n_{links}	Total number of segments in the partitioned input structure
$p(\cdot)$	Probability density function
r	Decay parameter in the Gaussian distribution
$\mathcal{S}$	Safe space
S^*	Safety of the optimal sequence $\mathbf{a}^*$
$S(\mathbf{a})$	Safety of the decision sequence $\mathbf{a}$
$S(a_k)$	Safety of the decision a_k
$S_d(a_k)$	Dynamic safety of decision a_k
$S_g(a_k)$	Geometric safety of decision a_k
S_k	Safety of the k^{th} decision
s_k	Environment state at decision step k
$T(t)$	Total kinetic energy at time t
$\tilde{T}(t)$	Normalized total kinetic energy at time t
$T_i(t)$	Kinetic energy of segment i at time t
t	Time
$[t_k, t_{k+1}]$	Time interval wherein motions begin and end
$\mathcal{U}$	Unsafe space
$U(t)$	Total gravitational potential energy at time t
$\tilde{U}(t)$	Normalized total gravitational potential energy at time t
$U_i(t)$	Gravitational potential energy of segment i at time t
$V(t)$	Set of points contained within all moving segments at time t
$V_i(t)$	Set of points contained within the moving segment i at time t
$\mathbf{x}$	Position in $\mathbb{R}^3$
λ_k	Decision weights (exponents) in the weighted geometric mean
λ_{wc}	Decision weight (exponent) of the worst-case decision
$\mathbf{1}_{\mathcal{S}}$	Indicator function of the safe space $\mathcal{S}$

This entire opus is respectfully dedicated to all those who give, help, teach, and care.

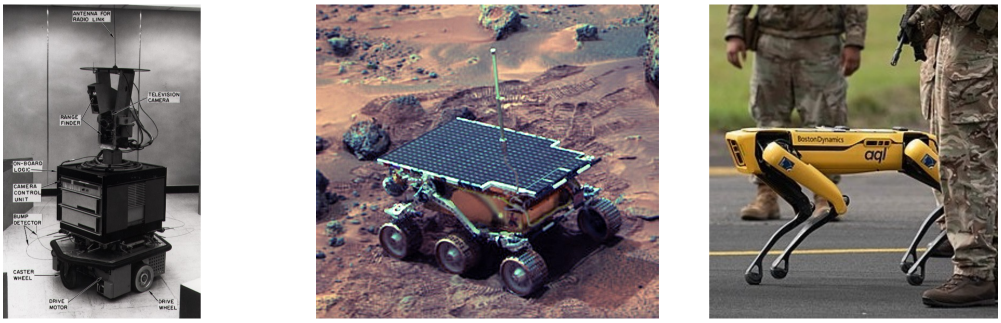
Chapter 1

Introduction

In the wake and aftermath of the Second World War, ‘machine cognition’ became a subject of growing interest [1]. The myriad of efforts and trends that followed gestated into the various disciplines we know today [2], for instance: artificial intelligence, control theory, cybernetics, data science, decision theory, robotics, among many others. Ever since, the sophistication of autonomous and intelligent systems has been growing at an ever-increasing pace. In effect, robotic systems have improved in virtually every aspect of their capability, intelligence, and autonomy enabling them to accomplish ever more complicated tasks in complex environments.

Initially, robots and agents were expected to operate mostly in narrow, focused, structured, or semi-structured settings. In recent times, however, the desire grew [3] towards these agents operating, surviving, and perhaps thriving within increasingly complex settings. Perhaps customarily, one can sketch a brief timeline (Fig. 1.2) showing the expansion of robotics into increasingly adverse spaces with an upwards trend of task complexity and autonomy:

- **1961:** The first industrial robot, the Unimate, is installed at General Motors [4]; with systematic tasks stored in a drum memory.
- **1972:** The autonomous robot Shakey (Fig. 1.1a) and A* search are developed at SRI International [5], enabling more general planning and execution of motions.
- **1989:** The Q-learning algorithm, a model-free reinforcement learning technique, is introduced by Christopher Watkins [6], enabling policy learning from data.



(A) “SRI Shakey with callouts”, provided by SRI International, licensed under CC BY-SA 3.0, dated 1972.

(B) “Sojourner on Mars PIA01122”, provided by NASA, in the public domain, dated 1997.

(C) “Spot robot Royal Air Force”, provided by Senior Airman John Ennis, in the public domain, dated 2021.

FIGURE 1.1: Three autonomous robots across generations.

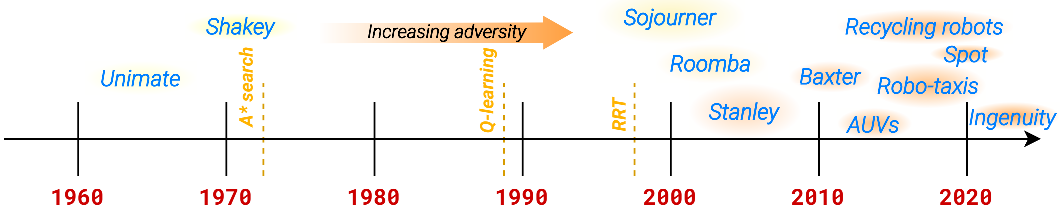


FIGURE 1.2: Timeline of accomplishments in the robotics community highlighting the increasing adversity within which robots operate.

- **1997:** The Sojourner rover (Fig. 1.1b), the first to autonomously operate on another planet, is landed on Mars [7].
- **1998:** The Rapidly-exploring Random Tree (RRT) motion planning algorithm is introduced by Steven LaValle [8].
- **2002:** The Roomba autonomous vacuum cleaner is introduced by iRobot to operate in domestic environments [9].
- **2005:** The DARPA Grand Challenge, a competition for autonomous vehicles, is won by the Stanford Racing Team for their robot, Stanley [10].
- **2012:** The Baxter robot, the two-armed collaborative industrial robot, is released by Rethink Robotics [11].
- **2014:** Autonomous underwater vehicles (AUVs) are used to search for the wreckage of the missing Malaysian Airlines flight MH370 in the Indian Ocean [12].
- **2016:** The first self-driving taxi trials begin in Singapore, demonstrating the potential of autonomous transportation in complex urban environments [13].
- **2017:** Recycling robots that can recognize and sort different recyclables are installed by AMP Robotics, Bulk Handling Systems, and ZenRobotics [14].
- **2020:** Boston Dynamics deploys the Spot robot (Fig. 1.1c) to help reduce the exposure of frontline healthcare workers to COVID-19 [15].
- **2021:** The robotic helicopter Ingenuity, carried by NASA's Mars rover Perseverance, makes the first powered flight on another planet [16].

In effect, the range and complexity of targeted environments have expanded significantly (Fig. 1.2). Despite their advancement in performance and scope, robotic systems and agents continue to struggle [17] within environments characterized by:

1. **Disorder:** Fewer presence of patterns, structure, or regularity;
2. **Variability:** Higher variance and uncertainty in the inputs and events;
3. **Uncertainty:** Abundant noise, disturbances, model mismatches, and unknowns;
4. **Complexity:** Overwhelmingly large state, action, or decision spaces; and,
5. **Volatility:** Highly-dynamic change and strong nonlinearities.

Against such adversity, rather than cementing deeper into one class of approaches, the advantages of multiparadigm thinking become more attractive [18] for the purpose of designing intelligent agents.

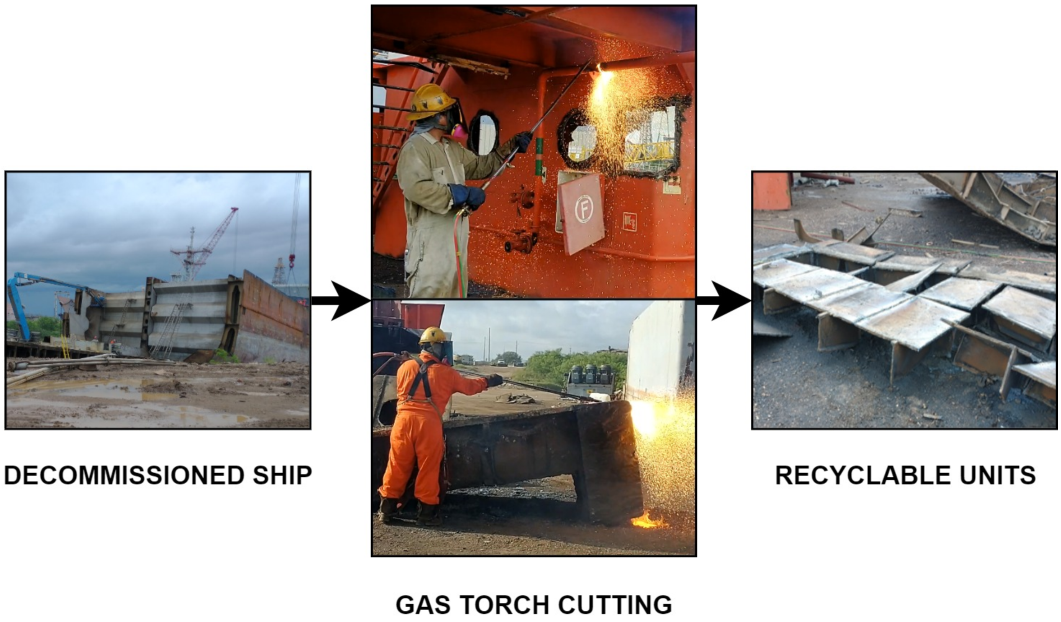


FIGURE 1.3: Shipbreaking operations break down a decommissioned vessel into salvageable parts, further cutting them into recyclable units. © IEEE

Such is one general theme of this dissertation: to explore the tackling of automation within highly adverse environments and to develop robot autonomy therein, through the examination and focus on the specific domain of metal scrap recycling. In doing so, this work provides a blueprint and example for the problem-solving and system design of autonomous and intelligent agents in environments of comparable adversity. More broadly, one of this dissertation's outcomes is to make explicit the advantages of implementing diverse cognitive architectures for enabling robots to challenge and exploit their environments' difficulties.

1.1 Dissertation Motivation

This work was motivated by the ambitious goal of automating key processes within the metal scrap recycling industry. The essence of this industry is the breakdown of large metallic structures (such as decommissioned vessels and aircraft) into smaller workable units of specified dimension (see Fig. 1.3). These recyclable units are then sold as scrap for their eventual processing into secondary metals (or recycled metals). In principle, the larger function of this industry is the material recovery of metals (having industrial value) from end-of-life or out-of-service structures and devices. In doing so, metal waste is reduced from landfills and oceans—recovered scrap is then processed into secondary (recycled) metals. More broadly, this industry provides employment opportunities to skilled workers and supplies (to its governments) the domestic disposal of retired military vessels.

However, this industry is characterized by work that is notably labor-intensive, hazardous, difficult, and notoriously resistant to mechanization and automation. In particular, the breakdown of these large structures—which in itself is dangerous—is accomplished manually using oxy-fuel cutting torches. In light of this, we give priority to the automation of oxy-fuel metal scrap cutting: generating a cutting path, controlling the torch's motions, monitoring for safety, and sequencing the cuts for safe structural disassembly.

This dissertation's motivation can thus be restated more precisely as: the design of a robotic system's autonomy and its associated collection of algorithms and methods to overcome the difficulties of the metal cutting operation in the scrapyard environment. By better taming some aspects of its tasks, our agent gets closer towards accomplishing its goals.

1.2 Dissertation Objectives

The overarching aim of this research is to provide a framework for robot autonomy to automate key tasks within metal cutting in the scrapyard environment. For this, the metal scrap industry is analyzed and characterized in order to understand the targeted tasks and their associated constraints, as well as the performance limitations of automating them.

Accordingly, this dissertation's specific goal lies in addressing each of these key tasks, consisting of the following:

1. Given an input scrapyard object, identify the path on its surface along which the robot must cut. These objects come in varying sizes and shapes, thus few prior assumptions can be made about their geometric and physical properties.
2. Given a desired cutting path, control the oxy-fuel torch's motion along this path to correctly and efficiently cut through the metal surface. The combustion along the path must be maintained at a desirable state.
3. Given images of an oxy-fuel cutting operation, infer the nominal cutting task state to validate actions or detect anomalies. The inference must be sufficiently efficient for online monitoring to improve the safety of the operations.
4. Given an input object, plan the sequence of cuts to safely break down the structure. This cut sequencing must account for the dynamic and geometric effects of object fragments falling around the cutter's location.

We thus present novel solutions to these domain-specific and application-driven problems. Our proposed automation framework integrates a variety of paradigms into a diverse cognitive architecture for endowing the robot with the autonomy to engage in its tasks within the scrapyard environment.

We follow these developments with an analysis of the broader impacts of this research (socio-technical, economic, and ethical) in terms of its projected benefits, concerns, and uncertainties. More broadly, this dissertation illustrates the advantages of diverse cognitive architectures for developing robot autonomy in highly-unstructured settings. Furthermore, it explores the application of robotics and automation to address an environmental problem, thus serving as an instance of *environmental robotics*.

1.3 Proposed Framework

To design autonomous systems for any complex environment, we must crucially begin with understanding, analyzing, and characterizing the problem and domain with great detail. This often requires on-site surveying and communication with experts from multiple disciplines. Without a substantial understanding of the domain, efforts

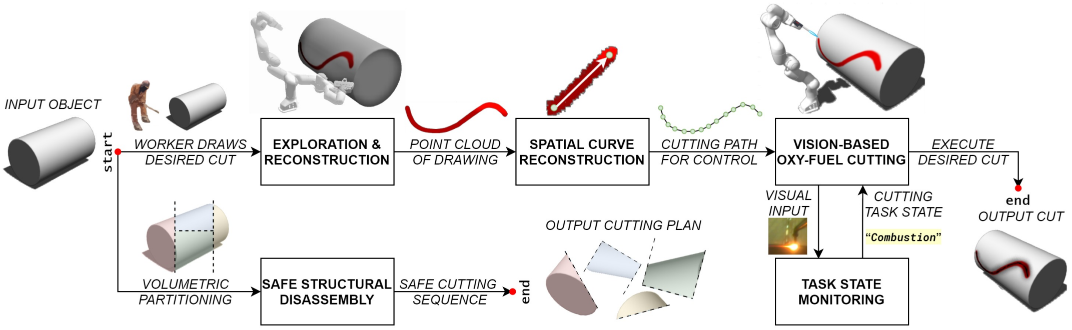


FIGURE 1.4: Conceptual overview illustrating the proposed automation framework and its components. The blocks show the methodologies that are developed in this dissertation.

in automation, representation, or modeling would risk solving a different problem, or risk tackling an overly simplified or idealized setting. In our case, we undertook an on-site survey in a representative shipbreaking yard to observe and study the key tasks in their real setting at their full complexity. With the insight and knowledge gained from surveying scrapping operations, and after learning from the workers' experiences and intuition, we deconstruct the problem and identify its most essential tasks that are achievable given the constraints in design and development.

Accordingly, we propose the following framework for the undertaking of automating the aforementioned key tasks within metal scrap recycling.

1. **Cutting path generation:** Obtain a reference cutting path on an input object's surface—using human–robot collaboration and autonomous viewpoint planning—without any *a priori* specification or knowledge about the shapes and sizes of the path or the object.

2. **Autonomous oxy-fuel cutting:** Operate an oxy-fuel cutting torch autonomously along a given reference cutting path—using a vision-based framework for calibration, surface conditioning, and control—while maintaining a desired combustion state on the metal's surface.

3. **Task state monitoring:** Infer the cutting task state from the robot's vision in order to monitor the validity of the robot's actions—using an a classifier designed for and trained on our image dataset curated from our numerous oxy-fuel cutting experiments—thus increasing the safety of the robot's autonomy.

4. **Safe structural disassembly:** Plan a cutting sequence algorithmically for disassembling large structures safely—using a mathematical model for safety, a physics-based simulation environment, and decision algorithms—given a volumetric partitioning of the input structure.

In this framework, the inputs, outputs, and requirements of the components as well as the interactions between them are summarized in Fig. 1.4. These are further elaborated in Chapter 3 along with their respective fundamentals.

1.4 Novel Contributions

To the best of our knowledge, the work covered by this dissertation consists of the first-ever research effort into automating the cutting operations within a metal scrap recycling yard. More explicitly, our novel contributions include:

1. A feature-driven next best view paradigm for exploring a subset of an unknown surface to reconstruct a feature of desired characteristic.
2. A vision-based framework for autonomous oxy-fuel cutting encompassing visual calibration, metal surface conditioning, and torch combustion control.
3. An oxy-fuel cutting task state monitoring strategy accompanied by a novel open-access image dataset usable for clustering and classification.
4. A cut sequencing algorithm for safely disassembling large structures via physics-based simulation and a mathematical model for safety.

1.5 Dissertation Structure

This dissertation is organized as follows. We first discuss in Chapter 2 the metal scrap recycling industry and its problems, opportunities and challenges. Next, we elaborate in Chapter 3 the specifics of our proposed framework and trace the fundamentals and methodologies of its components. In the chapters that follow, we develop each component in detail, specifically:

- Chapter 4 covers cutting path generation via human–robot collaboration and viewpoint planning. The object surface is explored and the desired cutting path is reconstructed using next view planning, next best view algorithms, and spatial curve reconstruction methods.
- Chapter 5 develops a vision-based control algorithm to regulate the motion of the oxy-fuel torch during combustion cutting. This requires the calibration of the vision system to reduce noise and retrieve the targeted features, as well as the conditioning of the metal surface for combustion cutting.
- Chapter 6 designs and trains a neural network-based classifier for online monitoring of cutting operations to infer the nominal cutting task state of the robot. For this, a dataset of oxy-fuel cutting footage is curated from live cutting experiments conducted using a 1-DOF robot.
- Chapter 7 devises a decision algorithm to disassemble partitioned structures safely for the cutter and the surrounding environment. For measuring the safety of a cut or a cutting sequence, the decision agent relies on the outcome of physics-based simulations and a mathematical model for safety.

Afterwards in Chapter 8, the broader impacts of this research are analyzed in terms of their projected benefits, concerns, and potential risks. Finally in Chapter 9, we conclude with the outcomes of the proposed framework and methods along with some insights and recommendations for future research.

Chapter 2

Metal Scrap Recycling

THIS chapter introduces the metal recycling industry and its operational contexts within shipbreaking and scrapyards. Therein, we characterize the scrapping environment and its operations. Accordingly, we examine the problems and corresponding opportunities of this industry. These are then assessed through the lens of automation and the benefits and challenges that follow. The operational insights and photographs in this chapter are derived from our on-site fieldwork consisting of surveying a representative large-scale shipbreaking yard in the Southern United States: EMR International Shipbreaking Limited, LLC based in Brownsville, TX.

Keywords: *Metal recycling, shipbreaking, metal scrapyards, on-site survey, scrapping operations, scrapyard environment, automation.*

2.1 Industry Background & Problems

The metal markets are sustained by the utilization of two different methods of material sourcing [19]: primary materials, which refer to virgin ores extracted from mines, and secondary materials, which refer to recycled metal scrap. In the case of steels, recycling steel scrap through the process of melting is a markedly more energy-efficient [20] method than producing primary steel from mined iron ores. To supply secondary materials for producing recycled metal, the metal scrapping industry carries out the breakdown of large decommissioned metal structures—such as vessels and aircraft—into smaller and workable recyclable units, later sold for the purpose of producing new and recycled metals.

2.1.1 The Shipbreaking Industry

A major stakeholder in metal recycling is the shipbreaking industry [21] which services the various marine industries seeking disposal of their decommissioned vessels. Bearing high overheads, these marine industries are especially vulnerable [22], [23] to global crises, market volatility, and uncertain market outlook. This is exemplified, for instance, during the fallout of the COVID-19 crisis when global supply chains constricted [22], [24] and fuel prices severely fluctuated. During such fluctuations, the servicing, maintenance, and operation of ships become more expensive endeavors, but more so, their future costs become more difficult to estimate. As a result, shipyards, construction companies, maritime shippers, and navies are faced with the difficult decision between maintaining their fleet, and selling off their most problematic ships for demolition and scrapping—especially when the options of refurbishment, storage, resale, or salvaging are unavailable, infeasible, or less attractive.

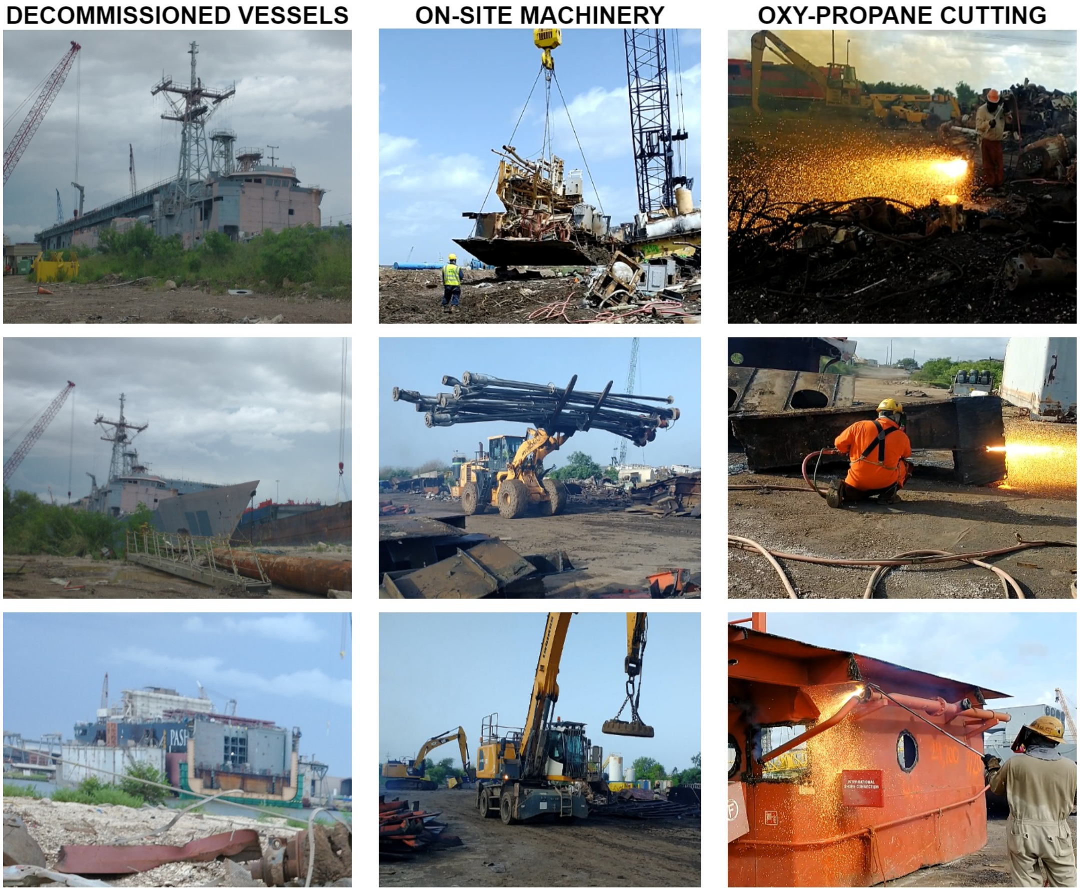


FIGURE 2.1: In the shipbreaking yard, the breakdown of decommissioned vessels is conducted via oxy-propane gas cutting and its ancillary logistics are facilitated using on-site machinery.

For companies, the decision of whether to maintain or scrap a vessel depends on their strategic opportunities, market conditions, and market outlook. Factors which support the scrapping of their vessels include [25] aging, depreciation, market shocks, rising commodity prices, and stricter regulations [26]. In the case of navies, an aging vessel can incur severe maintenance costs that increasingly burden government defense budgets, and by extension, the taxpayers. However, retiring warships such as aircraft carriers and battleships may constitute a breach of national security if acquired by foreign scrappers. Accordingly, government navies may give preference to domestic scrappers [27] even when their scrapping rates are less competitive. This is especially true when the vessels in question contain nuclear material, weapon systems, or exhibit classified engineering designs.

While there are a variety of ships that are subject to scrapping, the most common category that are recycled globally are transport vessels [28], such as bulk carriers, oil tankers, and container vessels. In effect, these transport vessels typically have 25–30 years of service [29] in the global marine shipping industry. In all cases, aging vessels that are decommissioned are typically sent to shipbreaking yards for recycling. During this scrapping process, the vessels are broken apart and then cut into recyclable metal components using gas torches (see Fig. 2.1 and Fig. 1.3).

2.1.2 The Scrapyard and its Operations

This industry is characterized by its labor-intensive and comparatively low-technology activity—shipbreaking operations are not straightforward to mechanize or automate. The working conditions in these scrapyards are difficult and dangerous, often requiring the disposal of hazardous substances [26], [30]. An essential operation in these scrapyards is metal cutting and its planning. Certainly, orchestrating the large-scale complexity of scrapping activities relies on many other crucial aspects, such as the operation of heavy machinery, logistics, transportation, and many others.

In the scrapyard, the most commonly used metal cutting medium is oxy-propane gas cutting. While the plasma cutting of metals is used within certain edge cases during scrapping, it is less compatible with the overall requirements of the scrapyard operations. In effect, each cutting medium has its own advantages and their comparative cost advantages [31] depending on the desired cutting thickness. As compared to oxy-gas cutting, plasma cutting can achieve (at a higher convenience) cleaner and more precise cuts at faster cutting speeds. However, unlike in manufacturing, these factors are unnecessary in scrapping. In manufacturing, a higher quality product leads to a higher added value and ultimately a higher price. By contrast, in scrapping the end-product is sold at commodity prices yielding tighter margins. Therefore, the cost-effectiveness of scrapping operations dictates the preference for the cutting medium. Moreover, plasma cutting would complicate logistics since it requires electrical shielding and smoke removal hoods.

When compared to plasma cutting, the preference for oxy-propane cutting is based on the following incentives: (1) propane is a relatively affordable fuel; (2) oxy-propane cutting penetrates carbon steel more deeply (albeit more slowly) and with a larger heat-affected zone; and, (3) carbon steel is the predominant recycled metal in shipbreaking yards, against which oxy-propane cutting is particularly advantageous. For these reasons, the bulk of the cutting operations are carried out using oxy-propane metal cutting via gas torches.

While scrappers may differ in their internal processes, we summarize a common pattern for scrapping operations [27]. First, the ships are examined and cleared of hazards, whether mechanical, chemical, or radioactive. Depending on the severity and complexity of the hazards, this may take months. Afterwards, safety forepersons and cutting workers enter the ships and assess the cutting scenarios. Typically, cutting plans are drafted and approved by safety forepersons prior to the execution of cutting operations. Next, cutting workers begin segmenting the large structural elements of the ship which are then lifted from the ship and placed on the ground by means of heavy machinery. These structures are then re-assessed by the forepersons and workers in newly drafted cutting plans. The worker then executes the cuts, which depending on the size of the structure, may span several days. Moreover, the worker cuts the structure into segments of approximate dimensions agreed-upon before cutting per market needs.

Finally, these recyclable units are sold to buyers of processed metal scrap. It is worth noting that depending on the targeted end-product, the specifics of these processes may vary. Downstream, metal scrap undergoes further processing which may include sorting, shredding, melting, decontamination, and solidification leading to the distribution of metal ingots. Furthermore, in the scrapyard, the prioritization of worker safety and well-being often varies widely between countries and employers. In

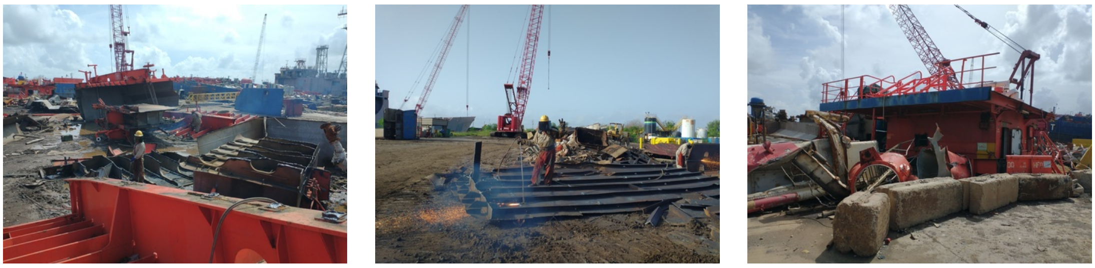


FIGURE 2.2: Metal scrapping operations are conducted in an outdoor shipbreaking yard.

poorly-regulated scrapping sectors, the impacts [32] on workers and the environment can be detrimental.

2.1.3 The Industry's Problems

Shipbreaking as an activity, an industry, a market, and a working environment suffers from a wide array of structural problems, outlined below.

Metal scrap cutting is laborious and under-mechanized.

Cutting steel and other common marine-grade metals and alloys—such as stainless steel, copper, aluminum, brass, and monel—is highly labor-intensive, dirty, loud, stifling, and dangerous work. The personal protective equipment (PPE) required on-site can be heavy and uncomfortable. This is especially true for shipbreaking activities concentrated within hot and humid climates such as in the Gulf of Mexico—where most cutting operations occur in the United States [26]. In less regulated scrapyards, workers may get substandard upfront training in oxy-propane metal cutting using gas torches and may often have to learn to operate dangerous tools through trial and error with little supervision. Moreover, the availability of low-cost labor often discourages companies from investing into capital-intensive mechanized solutions.

The work environment is harsh, unstructured, and dangerous.

The work often occurs in outdoor yards (see Fig. 2.2) which are exposed to the elements throughout every season of the year. Workers must mind their surroundings and terrain whose properties may vary—*e.g.*, muddy, slippery, soft, or hard—and must halt scrapping operations when conditions are highly adverse such as during rainfall. The work requires manipulating and lifting heavy loads as well as operating gas torches in stifling, enclosed, and sometimes dark spaces—as in the hulls and chambers of a decommissioned ship. Scrapyard conditions can often be highly hazardous, and work accidents [33] are not uncommon. In less supervised yards—especially those violating the many guidelines specified by the OSHA [34] for metal scrapping—occurrences of worker deaths [35], [36] are more likely. In addition, many wrecked ships may contain hazardous materials—chemical, biological, radiological, and nuclear (CBRN)—such as those inside oil tankers or nuclear-powered aircraft carriers. In some cases [27], on-site medical care and emergency provisions can be substandard, wherein even treatable accidents can be fatal.

American scrappers are overly dependent on U.S. government demand.

Due to the scrapyard's labor-intensive operations, labor costs often amount to at least half [26] of the total scrapping cost. A wide range of workers—supervisors, cutters, crane operators, truck drivers, and loaders—are needed for these large-scale operations. For especially complex vessels, these labor costs can reach up to 90% [26] of the total; especially when one factors in the cost of proper handling of hazardous materials. In the United States, labor wages are comparatively higher than the scrapping industries in developing countries. Accordingly, American scrappers are often less globally competitive in terms of cost and and are also more sensitive to fluctuations in commodity prices. In effect, decommissioned vessels can often be composed of steel content that makes up 90% of their mass or more [37], and accordingly their scrapping revenue would then heavily depend [38] on the prevailing market value of steel. Furthermore, the comparatively stricter government regulations enforced on American scrappers renders them less competitive (cost-wise) on the international market. In effect, the American shipbreaking sector relies heavily on scrapping decommissioned military vessels [27]. In order to uphold environmental standards set by the EPA [39], it is often necessary for the government to allocate substantial financial resources in support of domestic scrappers. As an illustration [27], the US government was required to expend a total of over USD 3 million in 2009 to undertake the cleansing and demolition of two cargo ships from the reserve fleet, equating to a cost of around USD 400 per ton. Furthermore, there have been occurrences of authorities reclaiming decommissioned ships in situations where the scrapping process failed to meet the necessary standards. As a result, the U.S. government constitutes a monopsony in shipbreaking, *i.e.*, being the largest source of market demand, thereby yielding a highly dependent domestic market. For these reasons, automation becomes increasingly attractive for its potential to attenuate many of these structural problems.

2.2 Automation: Opportunities & Challenges

Compared to many other industries which have been able to modernize and mechanize, the metal scrapping industry remains lagging requiring much of its hazardous and labor-intensive work to be accomplished manually.

2.2.1 The Need to Automate

While the metal recycling industry is vulnerable to market conditions, this is even more pronounced within shipbreaking given the industry's sensitivity to freight market conditions [40] and currency exchange rates [41].

Moreover, long-term projections indicate [42] that the global supply of scrapped steel from decommissioned vessels is expected to increase dramatically by 2050. This would impose drastic demands to upscale [43] the scrapping capacity of shipbreaking yards, which is limited [44] especially in industrialized countries [26] having comparatively higher labor costs. More generally, the efficiency of metal scrapping and of its processes would gain increasing importance since the demand for major metals in the 21st century is projected [45] to increase.

Many of these structural inefficiencies, risks, and market pressures can be alleviated through advancing automation within the industry. Indeed, both the productivity and the safety of metal recycling can be improved by integrating robotic and automated

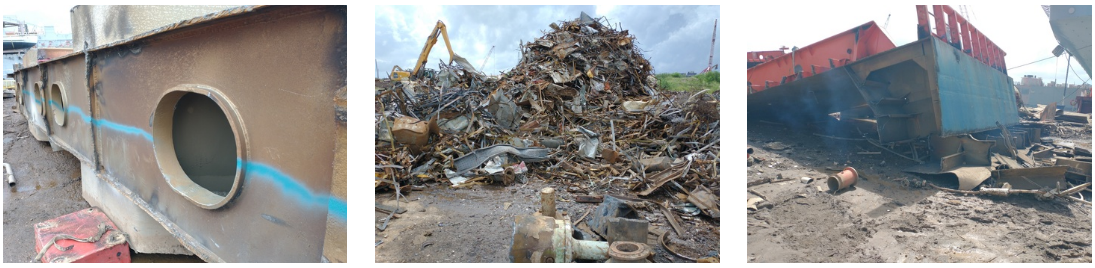


FIGURE 2.3: Scrap pieces in the shipbreaking yard vary greatly in size, shape, and geometric complexity.

systems—as is the case in various other sectors such as manufacturing [46] and logistics [47]. Such systems can target key processes such as metal sorting [48], but also crucially the metal scrap cutting operations. These would involve planning, executing, monitoring, and sequencing the oxy-fuel cutting process on diverse object structures of varying shapes and sizes.

2.2.2 Challenges towards Automation

While in many industries the task environment's structure and regularities can be exploited, conventional schemes for automation are ill-suited for the adversity of the scrapyard environment and the complexity of its operations. More precisely, the automation of metal scrap cutting in the scrapyard is challenging because of three salient factors:

The input objects to be cut are extremely diverse.

Scrap pieces in the scrapyard (Fig. 2.3) are highly variable, differing in properties such as size, shape, surface condition, spatial configuration, and so on. Objects can be small or large, simple or complex, smooth or sharp, visible or self-occluding, whole or perforated, among many other variations. Experienced workers can examine the cutting scenarios of a particular object to plan and execute cuts safely and efficiently.

The corresponding oxy-fuel cutting operations are complicated.

The kinematic parameters of cutting—such as the cutting speed along arbitrary cutting paths on arbitrary surfaces—are non-trivial to explicitly model and determine. This is in addition to accounting for combustion-specific parameters such as torch tip size and oxy-fuel flow proportions. In effect, combustion-cutting is a complicated process that skilled cutting workers learn intuitively by experimentation and supervision. Crucially, experienced cutting workers can identify anomalies and promptly stop the burning operations.

The environment hazards are difficult to assess.

This is especially true in the early stages of cutting, when structures are large and complex. The workers learn to select the desired cutting path on the object not only based on the object's properties, but also based on its surroundings and the predicted outcomes. An experienced worker and safety foreperson can assess the risks and hazards of particular cuts and agree on a cutting plan.

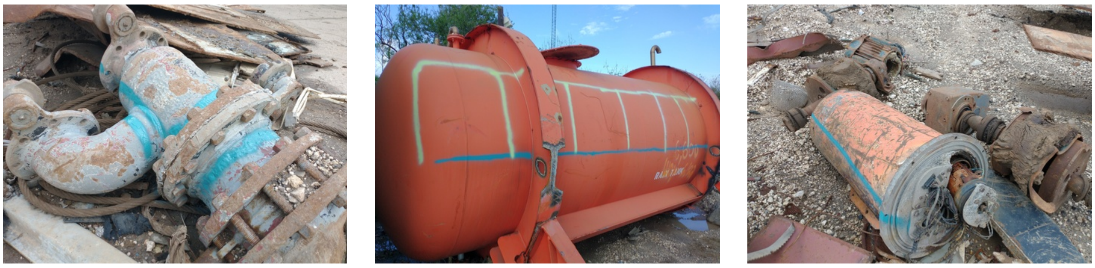


FIGURE 2.4: Three scrap pieces with desirable cutting paths on their surfaces painted by an experienced scrap cutting foreperson.

Thus, for each piece's individual properties and surrounding conditions, the desired cutting locations must be determined, the subsequent cutting trajectories must be generated, and the resultant cut must be executed with the adequate cutting parameters. In manual cutting, skilled workers can successfully determine the suitable operation objectives and can apply the adequate cutting parameters—all while compensating for disturbances and monitoring for safety. Translating all of these decision-making capacities and workers' implicit know-how into robot task parameters is highly challenging. While there are a few studies in the literature on robotic cutting, *e.g.*, [49]–[54], these focus on structured conditions, restricted tasks, controlled environments, or known object shapes. The development of robot autonomy for addressing the challenges of the metal scrap cutting environment is not examined in the literature.

2.3 On-site Survey: from Operations to Objectives

To develop fundamental contributions in environmental robotics that serve the practical applications of metal recycling, we must: (1) analyze and understand the targeted domain of metal scrap cutting; and, (2) enable the formulation of specific research problems. For this, we conducted on-site fieldwork consisting of closely surveying the operations of the aforementioned shipbreaking yard: EMR International Shipbreaking Limited, LLC based in Brownsville, TX.

These investigations resulted in the identification and specification of four salient research problems aimed at endowing robots with some autonomous functionality for metal scrap cutting. These problems are specified in order.

Cutting path generation

Since object pieces are highly diverse, it is required to identify an adequate path on the object's surface along which the cut is executed using a cutting torch. This desired path varies depending on the properties of the object and of its surface. For illustration, each of the three objects in Fig. 2.4 has a surface that is roughly cylindrical. Yet, the manner in which each of these is cut—indicated by the markings painted by a scrap cutting expert—can be significantly distinct due to the differences in size and features. Stated concisely, this problem is: given an arbitrary object surface, generate a cutting path. Equivalently, this can be stated as:

$$\text{`Input object'} \longmapsto \text{`Cutting path'}.$$

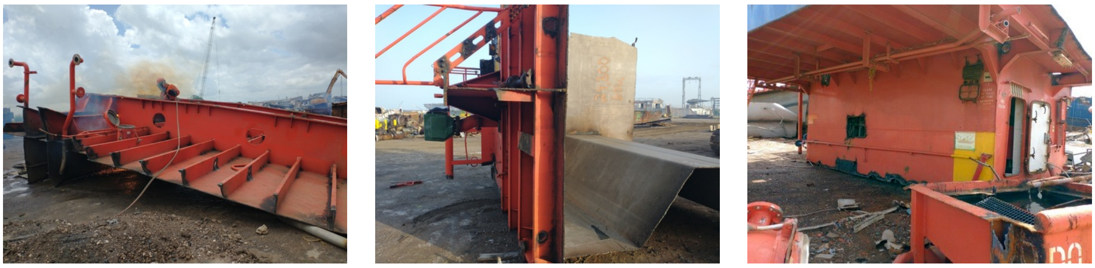


FIGURE 2.5: The shipbreaking yard processes large metal structures (shown in the images) by cutting them into smaller fragments.

Chapter 4 tackles this problem by first simplifying it via human–robot collaboration and then solving it using surface exploration, viewpoint planning, and spatial curve reconstruction.

Autonomous oxy-fuel cutting

After obtaining a desired cutting path, a robotic agent would need to execute it by means of an oxy-fuel cutting torch. This would involve controlling the torch motions along the path, *i.e.*, regulating the torch flame's tangential velocity along the path on the object surface. The problem can be summarized as:

$$\text{‘Cutting path’} \longmapsto \text{‘Cutting execution’}.$$

Chapter 5 addresses this problem by developing a vision-based framework for oxy-fuel torch control inspired by the visual workflow of skilled cutting workers.

Task state monitoring

Since the oxy-fuel cutting process itself is delicate but also prone to risks and hazards, it is desirable to monitor the task state of the operation and to halt upon detecting anomalies. Skilled cutting workers can do this instinctively and can aptly respond to anomalous events during cutting. Concisely stated, this is:

$$\text{‘Cutting execution’} \longmapsto \text{‘Cutting task state’}.$$

Chapter 6 targets this problem by curating an image dataset for oxy-fuel cutting and designing a CNN model for classifying cutting task states.

Safe structural disassembly

In the shipbreaking yard, the fragments separated from decommissioned vessels tend to be large metal structures as seen in Fig. 2.5 and are placed on the ground by means of heavy machinery for further cutting. The disassembly of these large structures exposes the cutting agent to a variety of hazards such as falling fragments, shifting centers of mass, collapsing structures, and so on. Prior to executing any cuts, skilled cutting workers typically agree with safety forepersons on a drafted cutting plan, which can often be revised during cutting if necessary. The goal is then to plan a sequence of cuts to fragment the object into smaller segments, that is:

$$\text{‘Input object’} \longmapsto \text{‘Cutting sequence’}$$

Chapter 7 targets this problem by designing a sequential decision-making algorithm for safely disassembling large structures.

The automation framework proposed in Chapter 3 is built on combining these four functionalities into a diverse cognitive architecture. Its purpose is to endow a robotic system with autonomous behavior towards facilitating the automation of metal scrap cutting operations.

Chapter 3

Proposed Automation Framework

We propose a novel framework for tackling the automation of metal scrap cutting operations. This is achieved by employing task-specific robot autonomy alongside worker expertise within a collaborative workflow. In this workflow, the skilled worker studies the scene containing the input scrap piece, determines an appropriate cutting reference, and marks it on the object with spray paint. The robot then autonomously explores the object's surface to identify and reconstruct the surface's painted regions from which a reference cutting path is obtained. Afterwards, the robot executes the cut along this desired cutting path on the object's surface by means of an oxy-fuel cutting torch using visual feedback. During execution, the robot's eye-in-hand vision is focused on the heated surface to monitor the cutting task state and detect the occurrence of anomalies in which case the operation is halted. More generally, the cutting sequence to safely disassemble a large structure can be planned using a special-purpose sequential decision-making algorithm.

Keywords: *Robot autonomy, metal scrap recycling, metal cutting, environmental robotics, field robots, human–robot collaboration.*

Notice: Part of the work in this chapter is published at the IEEE Transactions on Automation Science and Engineering (T-ASE) [55] and is subject to IEEE Copyright. An earlier version of this work is published at the IEEE 17th International Conference on Automation Science and Engineering (CASE) [56] and is subject to IEEE Copyright.

3.1 Scheme Overview

The complexity of the scrapyard environment and the complexity of its tasks—as discussed and analyzed in Chapter 2—can be addressed through: (1) reducing complexity by exploiting the implicit knowledge and experience of skilled workers; and (2) developing a orchestrated set of diverse functionalities, each addressing a specific task in the environment. In effect, by formulating and employing special-purpose methodologies tailored to the particularities of each task, parts of the environment can be tamed. In doing so, our collaborative workflow combines the respective strengths of both the robot and the skilled worker. The advantages of human–robot collaboration are commonly observed in industrial settings [57] and in the same spirit our scheme yields particular benefits:

- We leverage worker expertise that is difficult to transfer to the robot. The worker examines the scenario of cutting a particular object and provides cutting references to the robot by conveniently drawing them on the object's surface.

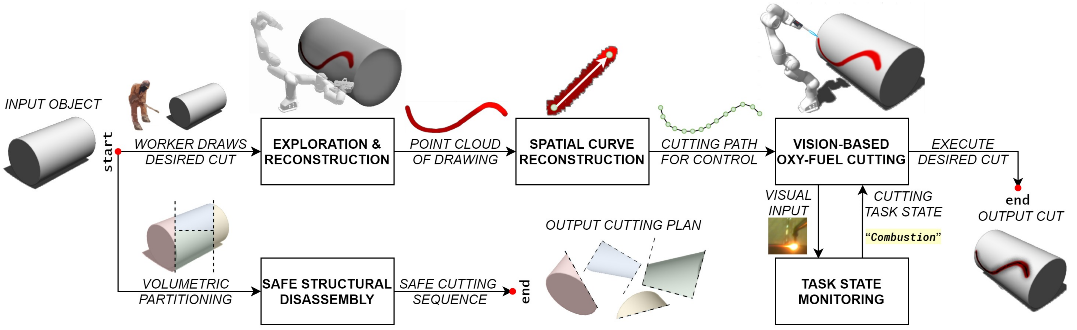


FIGURE 1.4: Conceptual overview illustrating the proposed automation framework and its components. The blocks show the methodologies that are developed in this dissertation.

- The work required by the robot is significantly reduced. The robot explores only the marked regions on the object—this avoids wasteful scanning of the entire object's surface.
- Accordingly, the robot can acquire the desired cutting path (per the worker's specification) and execute the cut using an oxy-fuel torch and visual feedback.
- The difficulty of the worker's labor is reduced as the tedious and high-risk tasks are delegated to the robot.

The proposed scheme's components are shown in Fig. 1.4 which is first presented in Chapter 1 and is reproduced in this section. The worker provides the cutting references on the object's surface in the form of drawings. Accordingly, the subsequent tasks for the robot are: (see Fig. 1.4)

1. Explore the surface and generate a cutting path from the drawing. (Chapter 4)
2. Executing the cut with the oxy-fuel torch using visual feedback. (Chapter 5)
3. Monitoring the cutting task state to detect any anomalies. (Chapter 6)

In addition, we develop in Chapter 7 a sequential decision-making algorithm to plan the cutting sequence for the safe structural breakdown of a large object. For this, the decision agent uses physics-based simulations and a mathematical model for safety to recommend a cutting sequence as well as prescribed cutter locations—where the cutter can be a human worker or a robotic agent.

For safe structural disassembly planning, this decision agent expects a volumetric partitioning (as in Fig. 1.4) on the input structure—wherein its segments and cutting location options are provided. In the scrapyard, this partitioning is informally performed during the planning by the safety foreperson and the worker (see Fig. 7.1) and can therefore be translated to the decision agent. Alternatively, Section 7.1 briefly reviews specialized algorithms for volumetric partitioning.

3.2 Robotic System Requirements

This framework is formulated for robotic systems of a particular configuration that is well-suited for the task requirements of metal scrap cutting. The hardware configuration of such robots would have to include a minimum set of actuation and sensing modalities suited for our framework. At the very least, this would consist of a robotic

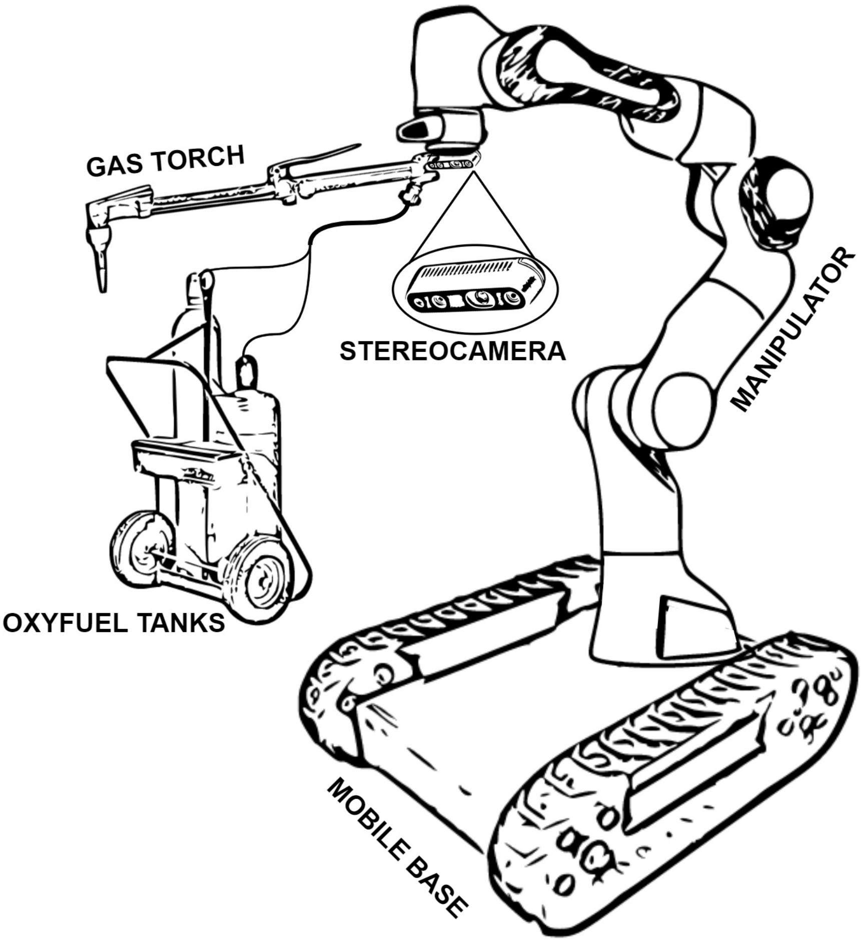


FIGURE 3.1: Conceptual diagram of the robotic system's hardware configuration. © IEEE

arm, RGB-D vision, and oxy-fuel cutting equipment. When mobility is desired in implementation for the scrapyard such a system would also need a mobile platform adapted for the adversity of the scrapping environment and its terrain. These are discussed and illustrated in Fig. 3.1 in an assumed robotic system.

3.2.1 Hardware Configuration

The robotic system that is conceptualized (see Fig. 3.1) for implementing our proposed framework consists of the following elements:

- **Robotic manipulator:** A serial arm mounted with a camera and cutting equipment, thus allowing it to explore its surroundings and execute cuts.
- **RGB-D camera:** This would be mounted eye-in-hand to provide the robot with visual feedback in the form of RGB images as well as 3-D colored point clouds.
- **Cutting equipment:** Most aptly, an oxy-propane cutting torch with appropriate fuel sources as well as logistical and safety provisions.
- **Mobile platform:** This vehicle would structurally support the aforementioned components but also enable mobility and navigation on the scrapyard terrain.

In our research, we employed specialized experimental setups to demonstrate and evaluate our methods. We do not explicitly implement a fully-fledged and production-ready field robot. We instead develop our framework and methods to enable such a system with autonomous functionalities. The specialized experimental setups used by each of our methods are clearly delineated in their respective chapters.

3.3 Review of the Underlying Methods

Before developing and detailing our methodologies in the subsequent chapters, we broadly and briefly review some of underlying methods and techniques used in our framework and indicate the chapter within which they are employed.

3.3.1 Viewpoint Planning

In many vision and perception applications, obtaining information from a single view may be insufficient or unreliable for performing the task. As such, the perceiving agent must select several viewpoints from which more information—and preferably more pertinent information—is sampled. The agent thus requires a mechanism, viewpoint planning [58], for selecting these viewpoints to assist its task, *e.g.*, scene exploration, environment mapping, object recognition, or object reconstruction.

Often, the viewpoint planner relies on a viewpoint sampling method to obtain a set of candidate views. Afterwards, the planner uses a viewpoint selection method to score these candidates and select that which optimizes for certain criteria. Sampling viewpoints can be accomplished deterministically such as in view sphere sampling where views are sampled on a discretized sphere centered around the object. Alternatively, sampling can be probabilistic such as with Monte Carlo viewpoint sampling. After obtaining a sample of viewpoints, the agent must select the next view to move towards. Selection can be based on optimizing some criteria, such as maximizing distance, visibility, or coverage or additionally by minimizing occlusions.

In performing viewpoint planning, the agent can face a variety of challenges, such as:

- Operating within uncertainty or partial observability—for instance when the surroundings are unknown or when objects can change locations or appearance.
- Balancing between accuracy and efficiency during planning—since capturing more details consumes more computational resources and processing time.
- Overcoming occlusions in the scene and self-occluding objects wherein certain details in the scene may not be visible to the agent.

In addition, there are iterative schemes for viewpoint planning, notably next best view (NBV) planning. NBV methods are used for selecting the viewpoint that maximizes a specified quality during each sequential planning step. The agent utilizes the currently available knowledge of its surroundings to select its next best viewpoint. This iterative process continues until the agent reaches some termination criteria, such as an accuracy threshold or an empty search frontier. The design of the quality to be maximized defines the behavior of the planner and its success. One common approach for defining quality is to use information gain formulations [59] whose aim is to maximize the expected information gained from the next view. Often, this is achieved by estimating the current knowledge's uncertainty, *i.e.*, information entropy, and by selecting the viewpoint which maximally decreases it. NBV algorithms have numerous applications, some of which are reviewed in Section 4.2.3.

In Chapter 4, we use viewpoint planning and next best view algorithms to enable the tasks of exploring surfaces and reconstructing feature point clouds.

3.3.2 Vision-based Control

Visual feedback can be a highly suitable sensing modality for some control tasks. In such cases, vision-based control schemes [60] use this visual information to estimate states and perform actions in the environment. Depending on the task and the environment, the visual feedback may contain complex information but may also be subject to noise and disturbances. Accordingly, such control schemes often rely a multitude of image processing techniques and other computer vision algorithms. The visual data is processed to extract useful features relevant to the control task as well as reducing noise. Such features can be any characteristic that can be detected and tracked in the visual data. Examples include shapes (*e.g.*, points, lines, edges, and corners), colors, or more complex features such as keypoints and robot joints.

The visual data may first undergo a noise filtering step to enable more robust feature detection, *i.e.*, identifying the features of interest. These features must be consistently identified and tracked across multiple frames. In some cases, this can be trivial as features may not change much across frames. When there are significant variations, however, then features must be matched, *e.g.*, by finding correspondences between them in different frames. This enables the estimation of change in the features, for instance, motion, color shifts, or shape changes. Finally, a control rule can be formulated using the tracked and matched features, for example, by computing the error between the desired features and the current features and using this error to adjust the control signals. Vision-based control can be quite flexible in its sensitivity to changes in the environment and its potential to handle a diversity of visual features. Nevertheless, the visual data must be appropriately processed to attenuate unwanted effects such as occlusions, noise, and variable lighting conditions.

In Chapter 5, we design a vision-based control algorithm that regulates the cutting speed of an oxy-fuel torch. This is achieved by computing (from the visual data) the combustion state of the resultant heat pool on the metal surface which can then be regulated towards a desired value.

3.3.3 Neural Network-based Classification

In machine learning, classification is a supervised learning task whose goal is to learn a map from the input data to output class scores—relative measures of correspondence to predefined classes, where the predicted class has the highest score. The classification problem requires a dataset whose instances are labeled with the desired output class. This data is then partitioned into a training set (to train the model for classification) and a testing set (to evaluate the model's performance).

A neural network model [61] is a parameterized representation of possible mapping functions, where the choice of mapping parameters define the map. Such neural network models can be 'trained' for classification tasks, meaning that their mapping parameters are optimized to minimize the error between predictions and training labels. This is accomplished using suitable optimization algorithms which iteratively adjust the model parameters during multiple passes over the training set. After training, the model can be used as a classifier to classify new input data. The classifier's performance is evaluated against the test set and is assessed through a variety of performance metrics such as accuracy, precision, recall, and F1 score.

Neural network models can have one of the many diverse neural network architectures. Such architectures refer to the type and structure of the model's computational elements, often described in terms of 'layers' and the maps between them. These affect the manner with which data is being represented and transformed from the input, through the latent space (intermediate representations), to the output. Most neural network models incorporate compositions of affine transformations and (typically nonlinear) activation functions. Network architectures may also include more particular elements. For instance, convolutional neural networks (CNNs) make use of convolution kernels (or filters) and are well-suited for many vision tasks [62].

In Chapter 6, we curate an image dataset for the problem of classifying task states in oxy-fuel cutting. Using this data, we develop and train a CNN model and evaluate its performance. Such a classifier would enable the monitoring of automated cutting operations to improve their safety.

3.3.4 Sequential Decision-Making

In sequential decision problems (SDPs) [63], a decision agent must select decisions (or actions) one step at a time. In these problems each decision affects the future outcomes (non-episodicness). The agent's objective is to select a decision sequence that maximizes a cumulative score across the decision steps. The number of steps may be finite, *e.g.*, may terminate when reaching a goal state, or may continue arbitrarily.

Typically, SDPs consist of the following problem elements (terminology can vary):

- **State space:** The set of all possible states of the agent and the environment, which may be fully-observable or partially-observable.
- **Action space:** The set of all possible sets of actions (or decisions) that are available to the agent for each state.
- **Transition function:** The environment dynamics that map the selected action given a current state to its outcome (the next state). These dynamics can be deterministic (*e.g.*, set of rules, dynamical system, physics-based simulations) or stochastic (*e.g.*, probability models, random processes).
- **Reward function:** This provides the agent with feedback on the quality or success of its decisions—essentially assigning a score to each state–action pair. This reward can be immediate, delayed, or uncertain.
- **Policy:** The map from states to actions that dictates the agent's action selection and represents their decision-making strategy. The policy can be explicit (*e.g.*, rules, heuristics) or it can be learned over time through trial and error.

SDPs can be approached with a variety of sequential decision-making techniques, such as search methods, Monte Carlo methods, and reinforcement learning methods.

In Chapter 7, we formulate the SDP of safe structural disassembly and design a decision agent to sequence cuts that safely disassemble the input structure.

Chapter 4

Cutting Path Generation

WITHIN our framework proposed in Chapter 3, this chapter focuses on the problem of obtaining the cutting path along which to cut a given input object. In metal scrapyards, the high variability of object surfaces poses significant challenges to identifying a path along which to cut the piece. We propose a novel collaborative workflow for cutting path generation that combines worker expertise with robot autonomy. In this workflow, the skilled worker studies the scene, determines an appropriate cutting reference, and marks it on the object with spray paint. The robot, then, autonomously explores the surface of the object for identifying and reconstructing the drawn reference, converts it to a cutting trajectory, and finally executes the cut. This chapter focuses on the surface exploration and cutting reference reconstruction tasks, which require appropriate next view planning (NVP) algorithms. We devise three NVP algorithms enabling the robot to explore and extract desired features from the scene, *i.e.*, the drawn reference, without requiring any *a priori* object model. Contrasting with global or feature-agnostic NVP algorithms, our approaches guide the robot via desired local features to increase the efficiency of the exploration. We evaluate our NVP algorithms against six categories of objects both in simulation and in physical experiments.

Keywords: *Viewpoint planning, next view planning, active vision, next best view algorithms, 3-D feature reconstruction, human–robot collaboration.*

Notice: The work in this chapter is published at the IEEE Transactions on Automation Science and Engineering (T-ASE) [55] and is subject to IEEE Copyright. An earlier version of this work is published at the 2023 IEEE International Conference on Automation Science and Engineering (CASE) [56] and is subject to IEEE Copyright.

4.1 Introduction

The work in this chapter is motivated by the need of extracting a desired cutting reference determined and drawn on the object by scrap yard workers. From the robot's perspective, it must explore and reconstruct the drawing, starting from an unknown scene containing an unknown object featuring an unknown drawing. We assume that an RGB-D camera is attached to the tool-tip of the robot, and the color of the drawn path is significantly different from the object's color. The goal of the robotic system is to explore the object surface to uncover the drawn path entirely without colliding with the object. The exploration algorithm must overcome complex object shapes and must be fast enough for practical use in scrap yards. This means conventional exploration (active vision) techniques are insufficient since they focus

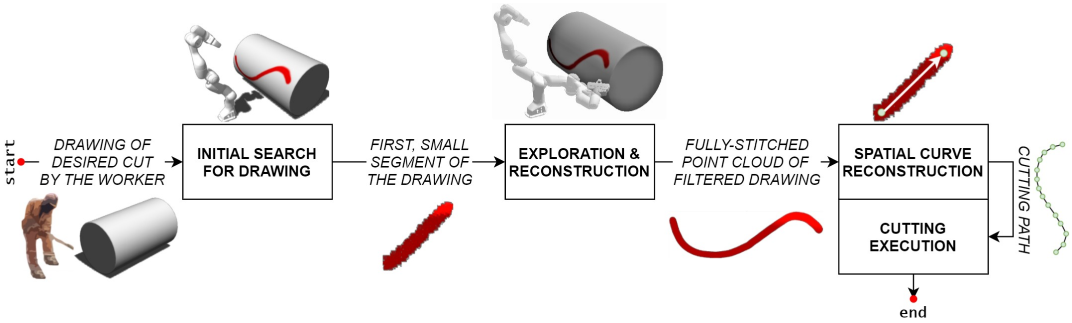

start
DRAWING OF DESIRED CUT BY THE WORKER
INITIAL SEARCH FOR DRAWING
FIRST, SMALL SEGMENT OF THE DRAWING
EXPLORATION & RECONSTRUCTION
FULLY-STITCHED POINT CLOUD OF FILTERED DRAWING
SPATIAL CURVE RECONSTRUCTION
CUTTING PATH
CUTTING EXECUTION
end

FIGURE 4.1: Block representation of the system's general workflow. This chapter's primary contribution concerns the "Exploration & Reconstruction" component. © IEEE

on exploring the entirety of the object, which is unnecessary for our task and is time-consuming. Our methods exploit the drawing information to guide the exploration for quickly determining a suitable viewpoint, which results in an efficient extraction of the entire cutting reference, without needing to explore the entire object's surface. Our algorithms are robust against adversarial features such as discontinuous, non-smooth, or self-occluded object surfaces. Our feature-driven strategies are not limited to robotic scrap cutting as they are applicable to any viewpoint planning problem requiring high performance while extracting the local features in the scene.

The proposed scheme's components are shown in Fig. 4.1.

4.1.1 Focus and Contribution

The scheme in Fig. 4.1 has two main tasks for the robot: extract the cutting reference and execute the cut. This chapter focuses on exploring the target object's surface and extracting the 3-D drawing to generate a desired cutting path.

It is important to note that the objects and drawings are both unknown to the robot prior to the operation, as they would be in scrapyards. Moreover, the objects in scrapyards are often large and complex in shape, which requires a systematic exploration of the objects' surface; the drawings would not be visible from a single viewpoint of the camera and are often times self-occluded. Therefore, the robot needs to scan the drawing one image at a time and reconstruct it piece by piece on a surface whose size and shape are both unknown. For this, the robot must carefully and intelligently plan its camera's next poses (viewpoints). This goal requires next view planning (NVP), *i.e.*, determining the camera's subsequent poses to fully explore the object's targeted feature. This chapter builds on our prior work [56], which outlined our human–robot collaboration scheme and presented our first NVP algorithm for cutting path reconstruction. While this algorithm was effective for objects with simpler geometries, it failed against more complex shapes by terminating prematurely. In this chapter, we address our prior method's shortcomings by adopting a next-best view strategy repurposed from the information gain formulations [64], [65].

This chapter accomplishes the following contributions:

- Propose a novel feature-driven active vision strategy to autonomously search for and reconstruct a drawn cutting reference on an object's surface, both having unknown size and shape.

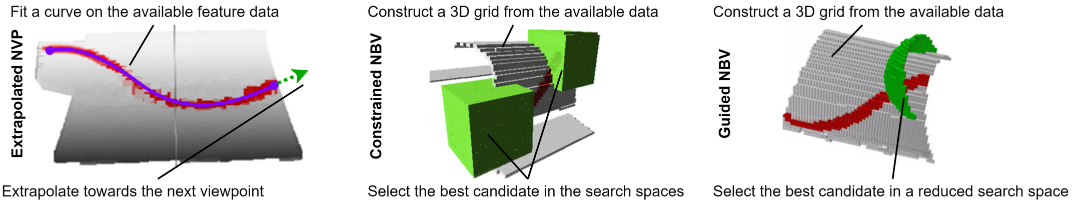

Fit a curve on the available feature data
Extrapolated NVP
Extrapolate towards the next viewpoint
Construct a 3D grid from the available data
Constrained NBV
Select the best candidate in the search spaces
Construct a 3D grid from the available data
Guided NBV
Select the best candidate in a reduced search space

FIGURE 4.2: Visual overview of the three proposed next view planning algorithm. "Extrapolated NVP" fits a curve to the feature points in the available data and obtains the next viewpoint pose by extrapolating from this curve while estimating surface normals. "Constained NBV" converts the available data into a volumetric occupancy grid, generates search spaces, scores every candidate viewpoint within, and then selects the candidate with the highest score. "Guided NBV" constructs the same grid but instead generates a reduced search space thereby rapidly obtaining a next viewpoint. © IEEE

- Develop two next best view (NBV) algorithms which respectively constrain and guide the search using feature information to select the camera's next viewpoint.
- Provide simulations and real robot experiments with six different object categories to evaluate the efficiency, flexibility, and robustness of our methods.

To the best of the author's knowledge, the work in this chapter is the first to develop cutting path generation for robotic cutting in unstructured scrapyards; and the first to present feature-driven active vision schemes for efficient and robust surface exploration. Our novel algorithms are not limited to scrap cutting but can also be applied to other problems that require 3-D feature reconstruction.

4.2 Related Work

This section, first, situates existing methodologies for robotic cutting and welding against our problem's challenges. Next, we discuss how our approach compares to current standards in active vision and NVP.

4.2.1 Cutting Path Generation for Robotic Cutting

As reviewed in [49], robotic cutting systems are employed in various industrial and medical applications. While these robots are highly diverse in their designs, objectives, and cutting media (*e.g.*, laser, plasma, oxy-gas, or water-jet), they often benefit from executing explicit and well-defined tasks in structured and controlled environments. For example, in manufacturing applications, object properties are known and can be directly modeled for the robot, as is the case for offline robot programming in [50] and [51]. Similarly, [52] improves the robot's cutting efficiency on known airframe parts in known locations to reduce manufacturing costs. For automated laser cutting [66], analytical methods in [67]–[71] are developed for known object geometries, while [72] improves path generation in a known and structured scene. In gas cutting, [54] develops a reactive control architecture for a robot to remove low-quality metal strips from sheet metal moved on a conveyor. Each of these uses contrasts with the operations of metal recycling scrapyards, where objects, their surroundings, and their cutting plans are unknown *a priori*. To the best of our knowledge, the only work on robotic shipbreaking is [53], where the focus is on applying a specialized cutting medium (hybrid induction plasma) to submarine recycling. This system operates only on submarine hulls, and therefore cannot handle the more difficult cutting operations in scrapyards. These formulations for robotic cutting are incompatible with our target application—metal scrap recycling—since they operate in structured environmental settings and directly on objects with full or partial knowledge of their properties. Instead, our algorithms explore and reconstruct an unknown drawing (cutting reference) on unknown objects. Our algorithms operate in different problem conditions, and so cannot be meaningfully compared against the aforementioned methods. We are not aware of any existing work on robotic cutting in unstructured environments—except ours in [56], which this chapter extends.

4.2.2 Weld Seam Extraction for Robotic Welding

While welding is a distinct operation from cutting, there are similar subtasks required for its automation. For example, the problem of weld seam extraction, which is covered in [73]–[79] has apparent similarities with retrieving a desired cutting path. Elsewhere, [80] proposes human–robot collaborative welding using a virtual reality setting. However, for scrap cutting, this approach limits the robot's task autonomy. It is worth noting that in welding, the goal is to join materials for the purpose of fabrication wherein there is abundant and reliable knowledge about the input objects. In effect, some of these seam extraction methods either assume partial or full knowledge of the seam, or expect a viewpoint containing it entirely. By contrast, the goal in metal recycling is to scrap or break down the objects into smaller workable units; noting that there is a large variety of objects and a high variance in their properties. Moreover, weld seams have favorable properties (precise, clear, and relatively predictable) which are often exploited to facilitate the seam extraction. In metal scrap cutting, however, the cutting references are noisy, irregular, and unpredictable. This is worsened by the fact that these drawings are painted by the worker in an unstructured setting. Moreover, scrapyard objects are often too large and complex to guarantee that the entire drawing can be fit in one view. For these reasons, these weld seam extraction methods are incompatible with our problem nor can they be directly compared with our algorithms.

4.2.3 Active Vision and Next View Planning

Active vision systems investigate the environment to gain pertinent information by manipulating the camera's viewpoint. Viewpoint planning has diverse applications, among which include precise manufacturing [81], shape reconstruction [82], human motion capture [83], underwater exploration [84], and robot grasping [85]–[87]. Recent developments also tackle less conventional configurations and solutions, such as exploring articulated scenes (manipulating surroundings while scanning) [88], scanning objects and scenes using an aerial robot [89], and estimating occlusions via surface edge exploration [90]. Learning-oriented paradigms are also found in NVP, most notably deep learning [91]–[93] and reinforcement learning [94]. These approaches are not suitable for our application since they do not exploit the relevant feature information to enhance the NVP. Instead, we avoid wasteful scanning of the entire object and target only its desired subsets.

Two of the algorithms proposed in this chapter are probabilistic NBV planners and use formulations similar to the information gain-based methods found in [64] and [65]. In these works, a probabilistic volumetric map is used for volumetric scene reconstruction, where the expected information gain is computed from discrete candidate views.

We emphasize that our NBV algorithms repurpose the aforementioned methods in a specialized framework adapted to the domain-specific task of robotic metal scrap scanning and cutting. While the formulations [64] and [65] are useful for global scene exploration and total object reconstruction, they are not tailored for nor are incompatible with our target application since they seek to map an entire object or scene and not the subset of the object surface containing the desired feature—thereby solving a distinct problem. In some cases, it may not even be possible—in our problem's formulation—to scan the object entirely due to workspace constraints within which our methods operate. Accordingly, we adapt their strategies into feature-driven planners by assisting the viewpoint search using the desired feature's information. To the best of our knowledge, no existing approach targets local features on unknown objects in unstructured scenes, a need which we address in this chapter.

4.3 Overview of the System

This section presents the different workflow components, the robot tasks, and the hardware configuration required for this chapter.

4.3.1 Hardware Configuration

We design our algorithms assuming that the system includes a mobile platform, a manipulator, an RGB-D camera, and a gas torch connected to appropriate oxy-fuel tanks (see Fig. 3.1). For the purposes of this chapter—both in simulation and in the physical experiments—the manipulator in use is the Franka Emika Panda 7-DOF arm and the RGB-D camera equipped is the Intel RealSense D435. The Panda robotic arm is used stand-alone, and its integration to a mobile platform is not addressed. The design and implementation of such a mobile robotic base is explored in [95].

4.3.2 Workflow Components

In this human–robot collaborative workflow (presented in Chapter 3), the worker and robot have well-defined roles and functions. From the robot's perspective, the worker provides the cutting references on the object. Afterwards, the robot must discover and recover these references and then carry them out. We emphasize that the robot does not know the locations of neither drawing nor object, nor does it assume their shape or size. It merely assumes their existence, and must autonomously discover and acquire any required properties. This is essential for scrapyard environments, as there is little regularity in shapes, sizes, and cuts.

After worker input, the robot's tasks become: (see Fig. 4.1)

1. Search to discover an initial partial view of the drawing.
2. Plan next views to gradually reconstruct the full drawing.
3. Generate a 3-D cutting path from this recovered drawing.
4. Execute the cut and using visual feedback.

The workflow can be viewed as an action sequence on the input (colored point clouds) to the desired output (successful cut). This is: scan (tasks #1 and #2), generate the cutting path (tasks #3), and execute the cut with visual feedback (task #4). The main contributions of this chapter is to solve tasks #2 (obtaining the drawing) and #3 (generating a cutting path). For this purpose, task #1 is assumed complete, *i.e.*, we assume that a (small) part of the drawing is visible to the robot from the initial view. Task #4 is addressed in Chapter 5.

4.4 Exploration & Reconstruction

This section introduces and explains the subroutines of the exploration and reconstruction task.

4.4.1 Task Overview

Referring to Fig. 4.1, the exploration and reconstruction task must facilitate the eventual generation of a cutting path. Instrumentally, it must provide an appropriate reconstruction of the desired feature (here, a 3-D colored drawing). This reconstructed drawing must cover as much of the original drawing as possible. Thus, the task's desired output is a high-coverage point cloud reconstruction of the drawing.

Due to the high variance of object sizes and shapes, the robot's initial view is unlikely to contain the entire drawing. Accordingly, the robot keeps exploring the object until it fully uncovers the desired feature. Thus, the exploration and reconstruction routine is formulated as a loop that terminates once the drawing is considered fully explored. Until this stop condition is met, the robot repeats the following steps:

1. Acquire a colored point cloud image.
2. Transform the acquired cloud to the fixed base frame.
3. Combine clouds from several views to reconstruct a surface containing parts of the drawing.
4. Segment the drawing region from the reconstruction.
5. Plan the next viewpoint using the extracted points.
6. Move to the next view while avoiding collisions.

In essence, we iteratively reconstruct a subset of the object's surface that contains the entire drawing, and then at each step, we use its available portion of the drawing to inform the next viewpoint. We terminate the search once this reconstruction is considered to contain the entire drawing, which is then extracted. These steps are elaborated in the rest of this section.

4.4.2 Point Cloud Processing

The robot acquires colored point clouds obtained from the variable camera frame. The surface containing the drawing is reconstructed by combining the point clouds obtained from multiple views. These clouds are first transformed to the fixed base frame and are then concatenated.

More formally, let $k \geq 0$ be the exploration's current step. Let ${}^{\mathsf{c}}C_k$ be the cloud obtained at step k with respect to the camera frame c. The camera's pose at step k is ${}^{\mathsf{b}}\mathrm{T}_k$ in the base frame b. These acquired clouds ${}^{\mathsf{c}}C_k$ are concatenated to iteratively expand the cumulative knowledge of the drawing.

Accordingly, let ${}^{\mathsf{b}}C_{\mathsf{total},k}$ be the cloud containing all the RGB-D information obtained thus far, *i.e.*, the cumulative cloud after step k expressed in the base frame. This can be expressed recursively for $k \geq 1$ as follows,

$$\begin{cases} {}^{\mathrm{b}}C_{\mathrm{total},0} = {}^{\mathrm{b}}\mathrm{T}_0{}^{\mathrm{c}}C_0, \\ {}^{\mathrm{b}}C_{\mathrm{total},k} = {}^{\mathrm{b}}C_{\mathrm{total},k-1} \cup {}^{\mathrm{b}}\mathrm{T}_k{}^{\mathrm{c}}C_k \end{cases} \tag{4.1}$$

The transformations ${}^{\mathrm{b}}\mathrm{T}_k$ map all clouds acquired at different steps k to the base frame for concatenation. The expression ${}^{\mathrm{b}}\mathrm{T}_k{}^{\mathrm{c}}C_k$ maps all of the cloud's member points from the camera frame at step k to the base frame.

The loop's next step is to segment the drawing from the current cumulative cloud ${}^{\mathrm{b}}C_{\mathrm{total},k}$. Let $\phi_\xi(\cdot)$ be the filtering function defined by its parameter ξ which determines its filtering behavior. In our setup, $\phi_\xi(\cdot)$ filters by color such that ξ is an admissible range of colors. Let ${}^{\mathrm{b}}D_{\mathrm{total},k}$ be the cloud representing the drawing such that $\phi_\xi : {}^{\mathrm{b}}C_{\mathrm{total},k} \longmapsto {}^{\mathrm{b}}D_{\mathrm{total},k}$.

With these first four subtasks defined, we summarize their inputs, outputs, and interaction in the pseudocode below.

```
define ProcessClouds(step $k$):
    ${}^{c}C_k$ ← AcquireImageAt(${}^{b}T_k$)
    ${}^{b}C_k$ ← Transform(${}^{b}T_k$, ${}^{c}C_k$)
    ${}^{b}C_{total,k}$ ← Concatenate(${}^{b}C_{total,k-1}$, ${}^{b}C_k$)
    ${}^{b}D_{total,k}$ ← Filter($\xi$, ${}^{b}C_{total,k}$)
    return (${}^{b}D_{total,k}$, ${}^{b}C_{total,k}$, ${}^{b}C_k$)
```

We implement some of these 3-D point cloud processing tasks using the Point Cloud Library [96].

We note the distinction between the clouds ${}^{\mathrm{b}}C_{\mathrm{total},k}$ and ${}^{\mathrm{b}}D_{\mathrm{total},k}$. The cloud ${}^{\mathrm{b}}C_{\mathrm{total},k}$ is the local surface reconstruction of the object, *i.e.*, the concatenation of all the acquired and transformed point clouds ${}^{\mathrm{b}}C_k$ at each step k. In contrast, the cumulative feature cloud ${}^{\mathrm{b}}D_{\mathrm{total},k}$ is the filtered version of ${}^{\mathrm{b}}C_{\mathrm{total},k}$ and thus retains only the feature points (in our case, points of red color) with all other points discarded.

4.4.3 Next View Planning

Until now, we have defined the acquisition and processing of point clouds at a particular viewpoint pose ${}^{\mathrm{b}}\mathrm{T}_k$. The initial viewpoint at $k = 0$ supposedly provided by the mobile search (see Fig. 4.1) is assumed given. Beyond this, we must obtain the subsequent viewpoints to gradually explore and reconstruct the drawing. For this, we require a next view planner.

Conceptually, the NVP algorithm performs higher-level reasoning on the raw point clouds obtained from the `ProcessClouds`$(\cdot)$ procedure. Specifically, the NVP generates candidate viewpoints using information from the cumulative feature cloud ${}^{\mathrm{b}}D_{\mathrm{total},k}$, the local surface cloud ${}^{\mathrm{b}}C_{\mathrm{total},k}$, and the latest transformed cloud ${}^{\mathrm{b}}C_k$. The pseudocode below conceptually sketches the exploration and reconstruction routine.

```
define ReconstructDrawing():
    ${}^{b}\mathrm{T}_0 \leftarrow$ InitialViewpoint()
    for $k = 0, 1, \ldots, k_{\mathsf{stop}} - 1$:
        $({}^{b}D_{\mathsf{total},k},\ {}^{b}C_{\mathsf{total},k},\ {}^{b}C_k) \leftarrow$ ProcessClouds($k$)
        ${}^{b}\mathrm{T}_{k+1} \leftarrow$ NextViewpoint(${}^{b}D_{\mathsf{total},k}, {}^{b}C_{\mathsf{total},k},\ {}^{b}C_k$)
    ${}^{b}D_{\mathsf{total},k_{\mathsf{stop}}} \leftarrow$ ProcessClouds($k_{\mathsf{stop}}$)
    return ${}^{b}D_{\mathsf{total},k_{\mathsf{stop}}}$
```

The loop's stopping condition is determined and checked by the viewpoint planner, terminating the exploration at some eventual step k_{stop}. After termination, the cumulative cloud ${}^{b}C_{\mathsf{total},k_{\mathsf{stop}}}$ should contain the entirety of the desired feature. The final and fully reconstructed drawing's point cloud ${}^{b}D_{\mathsf{total},k_{\mathsf{stop}}}$ is outputted for cutting trajectory generation.

Each of our three planning algorithms implement the ReconstructDrawing() routine with their own respective mechanism for NextViewpoint($\cdot$), *i.e.*, planning the next viewpoint. Irrespective of choice, the viewpoint planner controls these two decisions during exploration:

1. *Termination*: Determine if the drawing is fully explored, and accordingly either proceed searching or terminate.
2. *Viewpoint Generation*: Provide and select candidate camera poses to continue the robot's search.

For scrap cutting, the NVP algorithm is subject to performance constraints. An inefficient planner slows down the exploration routine, which would worsen cutting productivity. The planning time is affected jointly by the number of steps and by the step duration. This often comes with a trade-off, as planners that finish with less steps tend to spend more time per view, and by contrast planners which compute steps rapidly tend to iterate more. This trade-off is present in our algorithms and is later examined in our evaluations.

The NVP algorithm determines poses to visit sequentially, throughout the exploration task, to reconstruct the drawing on the object surface. The motion planner attempts to plan a feasible trajectory towards these poses and executes the first viable one. This motion then executes while avoiding collisions. For our implementation, we use ROS [97] and Gazebo [98]. We also use MoveIt! [99] for motion planning, which is internally set to use OMPL and TRAC-IK.

4.5 Next View Planning Algorithms

This section develops the three feature-driven NVP algorithms. Each algorithm:

- Obtains a point clouds of the object that is marked with the cutting path.
- Segments the points that belong to the cutting path.
- Processes the object point cloud and the cutting path point cloud to calculate the next viewpoint.
- Moves the robot to the next viewpoint, and repeats this process until a termination condition is reached.

The algorithms can be distinguished (see Fig. 4.2) by how each of these three high-level actions is accomplished.

The first algorithm generalizes the procedure for point cloud fitting and curve extrapolation, from our prior work [56]. To the best of the authors' knowledge, this method is the first of its kind to address the application of interest: reconstructing an unknown drawing on an unknown surface, given a known feature (here, the color red). As such, this algorithm is used as a baseline against which the remaining two are compared.

The second algorithm reformulates the methodologies in [64] and in [65] into one that is adapted for the application of interest. Briefly, the method relies on a probabilistic occupancy voxel grid and a quality metric (based on information gain) to explore and rank candidate viewpoints, thereby selecting that which maximizes quality. This planner searches the grid in a rather exhaustive manner. In contrast, our third algorithm, which uses the same formulation as the second, exploits greedy-like optimizations for faster searching.

We examine each method in its respective subsection.

4.5.1 Extrapolated Next View Planning (E-NVP) Algorithm

This first planner treats the feature NVP task like a path exploration problem—by iteratively exploring the branches of an unknown path until all endpoints are found. The point clouds are converted to more useful and more structured representations using fitting methods. The next view is obtained by extrapolating those fits.

Exploring along a path requires a sense of its direction. However, the data obtained from the stereocamera, which is then processed in the procedure $\texttt{ProcessClouds}(\cdot)$, remains in the format of point clouds. Essentially, this is an unordered list of 3-D colored points where the direction of the drawing cannot be directly inferred. In this form, viewpoint planning on the desired feature is not straightforward.

This planner solves this representation problem by mapping the point cloud to a more usable structure. The cloud data is used to construct or fit spatial curves parametrized in 1-D. This procedure is captured by the map $\mathbb{R}^6_{\mathsf{xyzRGB}} \to \mathbb{R} \times \mathbb{R}^3_{\mathsf{xyz}}$ where the input space $\mathbb{R}^6_{\mathsf{xyzRGB}}$ represents the point cloud. The output space (a curve) can equivalently be rewritten as a rule $\mathbb{R} \to \mathbb{R}^3_{\mathsf{xyz}}$, which takes a 1-D parameter and returns a 3-D point. The cloud is thus reduced to a curve as follows,

$$\left\{ \left({}^{\mathsf{b}}p_{\mathsf{x}}, {}^{\mathsf{b}}p_{\mathsf{y}}, {}^{\mathsf{b}}p_{\mathsf{z}}, p_{\mathsf{R}}, p_{\mathsf{G}}, p_{\mathsf{B}} \right)_j \right\}_{j=1} \xmapsto{\texttt{Fit}} \left({}^{\mathsf{b}}\mathbf{q}_{\mathsf{xyz}}(t) \right)_t \tag{4.2}$$

This 1-D ordering of 3-D points yields a sense of direction along the curve wherein there is an ordering on the curve points. Specifically, the set of unordered points $\{{}^{\mathsf{b}}\mathbf{p}_j\}_{j=1}$ is used to create the list of ordered points $\left({}^{\mathsf{b}}\mathbf{q}_{\mathsf{xyz}}(t)\right)_t$ indexed by the parameter t. This provides the planner with a way to determine the next viewpoint to continue exploring the drawing. Since the curve represents one continuous and known portion of the drawing, the drawing's unknown regions come after the endpoint of this curve. This planner assumes that the entire drawing has two extremities, meaning it is branchless. Thus, after the initial viewpoint, there are two scenarios for curve endpoint selection. If the initial viewpoint happened to already contain one extremity of the drawing, then the planner explores in the direction of the second curve endpoint. On the other hand, if the initial view contains an intermediate portion of the drawing, then the planner has two candidate directions to explore. The curve endpoint closest to the camera is chosen to reduce movement time between unexplored endpoints.

With a chosen curve endpoint, the planner extrapolates to the next viewpoint at each step at a configurable distance $\delta_{\mathsf{distance}}$. When $\delta_{\mathsf{distance}} = 0$, the extrapolated endpoint itself becomes the next viewpoint. Exploring one curve endpoint at a time is slower, but plans towards the drawing extremity more conservatively. Alternatively, $\delta_{\mathsf{distance}} > 0$ places the next viewpoint, beyond the curve endpoint. This speeds up exploration but may cause some overshoot. Accordingly, keeping the extrapolation distance small is preferable.

Extrapolation only determines the position ${}^{\mathrm{b}}\mathbf{d}_{k+1}$ of the next viewpoint. To fully provide the next pose ${}^{\mathrm{b}}\mathrm{T}_{k+1}$, the planner must also determine the next orientation ${}^{\mathrm{b}}\mathrm{R}_{k+1}$ of the camera at that position ${}^{\mathrm{b}}\mathbf{d}_{k+1}$. The planner sets the orientation to be orthogonal to the surface at the viewpoint position ${}^{\mathrm{b}}\mathbf{d}_{k+1}$. This orientation is obtained by estimating the vector normal to the surface surrounding the viewpoint position ${}^{\mathrm{b}}\mathbf{d}_{k+1}$. This region is contained in the cumulative cloud ${}^{\mathrm{b}}\mathrm{C}_{\mathsf{total},k}$. Orienting the camera orthogonally helps the robot obtain a more accurate capture of the surface and thus avoid leaving gaps in the drawing point cloud during exploration—which would otherwise require backtracking and cost additional time. The orientation about the normal does not affect output significantly, and is relaxed for motion planning.

This planner continues searching by iteratively obtaining new clouds, fitting a curve on the desired feature, and extrapolating towards the next viewpoint. The planner considers a drawing extremity to be found when the size difference between two consecutive reconstructions ${}^{\mathrm{b}}D_{\mathsf{total},k}$ falls below a threshold δ_{size}. After both extremities are found, the search terminates and returns the reconstruction ${}^{\mathrm{b}}D_{\mathsf{total},k_{\mathsf{stop}}}$. This iterative extrapolation procedure is outlined in Algorithm 1, where the result is the feature's reconstruction.

Note that this procedure is affected by the fitting method choice. In our prior work [56], we examined and compared the effects and results of two types of fitting methods. The first fitting method relies on an optimization scheme to fit a non-uniform rational B-spline (NURBS) curve to the cloud. This method initializes a proposed curve and then improves the fit using point-distance minimization. An alternative fitting method from computational geometry is topological skeletonization. The goal of a skeletonization algorithm is to compute the medial axis of a shape—defined as the set of points equidistant from its boundary. This discretizes the shape into a voxel grid, and then erodes its edges away by repeatedly applying a thinning algorithm, until a single voxel-wide skeleton remains. This method runs faster than the NURBS-based method, and is used in our evaluations.

4.5.2 Constrained Next Best View (C-NBV) Algorithm

This planner restructures the NVP task into a search problem on a voxel grid. In brief, this grid is constructed from the point clouds and updated at each measurement. A frontier region is computed to constrain and generate the candidate search space, within which each candidate is scored with a viewpoint quality metric. The best candidate is selected as the next view. This repeats until a termination criterion is met.

Algorithm 1: Extrapolated Next View Planner (E-NVP)

```
Input   : Initial view containing part of the drawing.
Output: ${}^{\mathrm{b}}D_{\mathrm{total},k_{\mathrm{stop}}}$, fully-reconstructed drawing cloud.
Initialize step, extremities found, and viewpoint.
  step $k \leftarrow 0$
  $n_{\mathrm{extremities}} \leftarrow 0$
  ${}^{\mathrm{b}}\mathrm{T}_0 \leftarrow$ InitialViewpoint()
Explore the drawing until two extremities are found.
while $n_{\mathrm{extremities}} < 2$ do
    Acquire, transform, concatenate, and filter clouds.
      $({}^{\mathrm{b}}D_{\mathrm{total},k},\ {}^{\mathrm{b}}C_k) \leftarrow$ ProcessClouds($k$)
    Backtrack to initial view if drawing size unchanged.
    if size$({}^{\mathrm{b}}D_{\mathrm{total},k})$ − size$({}^{\mathrm{b}}D_{\mathrm{total},k-1}) \leq \delta_{\mathrm{size}}$ then
        $n_{\mathrm{extremities}} \leftarrow n_{\mathrm{extremities}} + 1$
        ${}^{\mathrm{b}}\mathrm{T}_{k+1} \leftarrow$ InitialViewpoint()
    else
        Fit a curve on the cumulative drawing's cloud.
          Curve ← Fit(${}^{\mathrm{b}}D_{\mathrm{total},k}$, FittingMethod)
        Obtain next view pose from fit's extrapolation.
          ${}^{\mathrm{b}}\mathbf{d}_{k+1} \leftarrow$ Extrapolate(Curve, $\delta_{\mathrm{distance}}$)
          ${}^{\mathrm{b}}\mathrm{R}_{k+1} \leftarrow$ GetNormalVec(${}^{\mathrm{b}}\mathbf{d}_{k+1}$, ${}^{\mathrm{b}}C_{\mathrm{total},k}$)
          ${}^{\mathrm{b}}\mathrm{T}_{k+1} \leftarrow ({}^{\mathrm{b}}\mathbf{d}_{k+1},\ {}^{\mathrm{b}}\mathrm{R}_{k+1})$
    $k \leftarrow k + 1$
return ${}^{\mathrm{b}}D_{\mathrm{total},k_{\mathrm{stop}}}$
```

The algorithm voxelizes the obtained processed clouds into an octree grid of occupancy probabilities. The grid distinguishes between occupied, unoccupied, and unknown cells based on their occupancy probabilities—respectively more than, less than, and equal to 0.5. We use OctoMap [100] as a probabilistic voxel occupancy grid. This grid allows efficient storage and querying of cell probabilities.

Let $\mathcal{G}_k$ be the voxel grid generated at step k. The local scene's cumulative cloud ${}^{\mathrm{b}}C_{\mathrm{total},k}$ is voxelized to map the currently available knowledge, while the reconstructed drawing's cumulative cloud ${}^{\mathrm{b}}D_{\mathrm{total},k}$ is used to identify those voxels belonging to the drawing. Since the drawing is the region of interest, we determine its frontier on the grid. Here we define a frontier cell to have at least one unknown neighbor and at least another neighbor belonging to the drawing's region. The frontier at step k is denoted by $\mathcal{F}_k$ and represents the boundary of current knowledge about the drawing used to determine high-vantage locations for generating viewpoint search spaces. This constrains the search at step k for the next view in search space $\mathcal{S}_k$ where candidate viewpoints are scored and ranked. The space is constructed by generating regions from geometric primitives (*e.g.*, cubes, spheres) around each frontier cell, and then concatenating them such that $\mathcal{S}_k = \bigcup_{f \in \mathcal{F}_k} S(f)$ where $S(\cdot)$ generates a search space primitive for a single cell.

The camera's viewpoint s_k can be expressed in the base frame as ${}^{\mathrm{b}}s_k = ({}^{\mathrm{b}}x, {}^{\mathrm{b}}y, {}^{\mathrm{b}}z, \alpha_x, \alpha_y, \alpha_z)$, where $({}^{\mathrm{b}}x, {}^{\mathrm{b}}y, {}^{\mathrm{b}}z)$ is the position in the grid, and $(\alpha_x, \alpha_y, \alpha_z)$ is the orientation obtained as anticlockwise rotations about the respective axes. It can also be expressed in the camera frame as ${}^{\mathrm{c}}s_k = ({}^{\mathrm{c}}x, {}^{\mathrm{c}}y, {}^{\mathrm{c}}z, \alpha_{\mathrm{R}}, \alpha_{\mathrm{P}}, \alpha_{\mathrm{Y}})$ where $({}^{\mathrm{c}}x, {}^{\mathrm{c}}y, {}^{\mathrm{c}}z)$ is the position with respect to the camera's frame, and $(\alpha_{\mathrm{R}}, \alpha_{\mathrm{P}}, \alpha_{\mathrm{Y}})$ is the orientation specified with respect to the local roll-pitch-yaw frame $(\widehat{\mathsf{R}}, \widehat{\mathsf{P}}, \widehat{\mathsf{Y}})$ about the camera's body (see Fig. 4.3). The transformations from ${}^{\mathrm{c}}s_k$ to ${}^{\mathrm{b}}s_k$ relate these expressions.

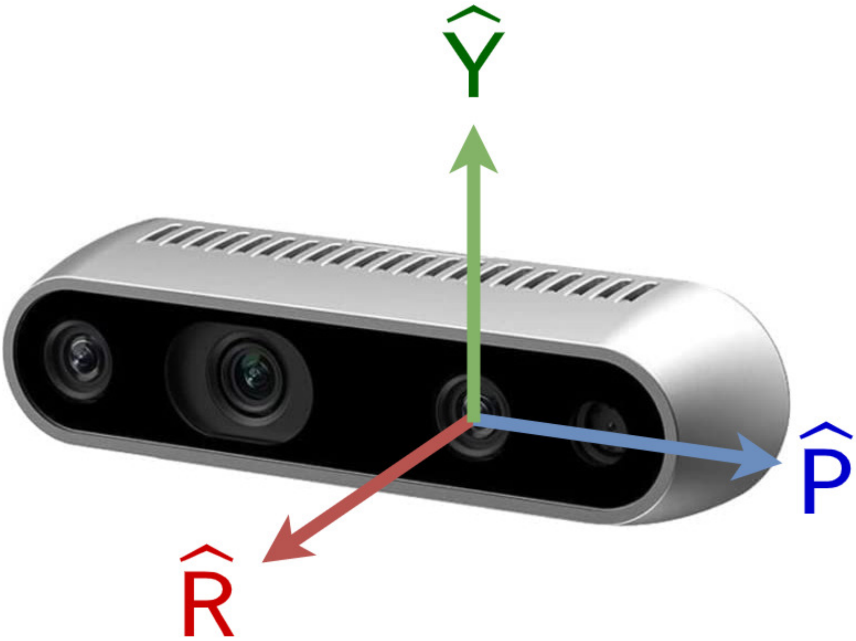


FIGURE 4.3: Specification of the camera frame $(\widehat{\mathsf{R}}, \widehat{\mathsf{P}}, \widehat{\mathsf{Y}})$ for orientation. $\widehat{\mathsf{R}}$ is normal to the lens extending from the front. $\widehat{\mathsf{P}}$ extends horizontally from left to right on the camera's front body. $\widehat{\mathsf{Y}}$ is obtained binormally to the rest. © IEEE

All candidates in $\mathcal{S}_k$ are scored using a viewpoint quality metric $Q(\cdot)$. The next view is set to the best-scoring candidate,

$$ {}^{\mathrm{b}}\mathrm{T}_{k+1} = \underset{s \in \mathcal{S}_k}{\operatorname{argmax}}\, Q(s) \tag{4.3} $$

We note that ${}^{\mathrm{b}}s_k$ represents the candidate viewpoints s in the base frame at step k of which the highest-scoring candidate $s^* = \operatorname{argmax}_{s \in \mathcal{S}_k} Q(s)$ becomes the chosen next viewpoint ${}^{\mathrm{b}}\mathrm{T}_{k+1}$ to which the robot moves from steps k to $k+1$.

The quality of an candidate viewpoint at step k is given by,

$$ Q(s_k) = \lambda \cdot \frac{\operatorname{gain}(s_k)}{\sum\limits_{s \in \mathcal{S}_k} \operatorname{gain}(s)} - (1 - \lambda) \cdot \frac{\operatorname{cost}(s_k)}{\sum\limits_{s \in \mathcal{S}_k} \operatorname{cost}(s)} \tag{4.4} $$

This is a convex combination of a gain term and a cost term, expressed as a proportion of their search space totals. The parameter $\lambda \in [0, 1)$ determines, for a particular viewpoint, the relative weight between its gain and cost terms $\operatorname{gain}(s_k) / \sum_{s \in \mathcal{S}_k} \operatorname{gain}(s)$ and $\operatorname{cost}(s_k) / \sum_{s \in \mathcal{S}_k} \operatorname{cost}(s)$, expressed as fractions of the candidate population totals—that is, the aggregate values for every $s \in \mathcal{S}_k$. Here, $\operatorname{gain}(s_k)$ quantifies the relevant information gain obtained from choosing s_k as the next viewpoint. The function $\operatorname{cost}(s_k)$ offsets this gain and is the Euclidean distance between the current viewpoint and the candidate s_k. This penalizes the quality of a viewpoint on distance from the current position. Therefore, a smaller value for λ prioritizes the highest-ranking viewpoint candidates which are also near the current position.

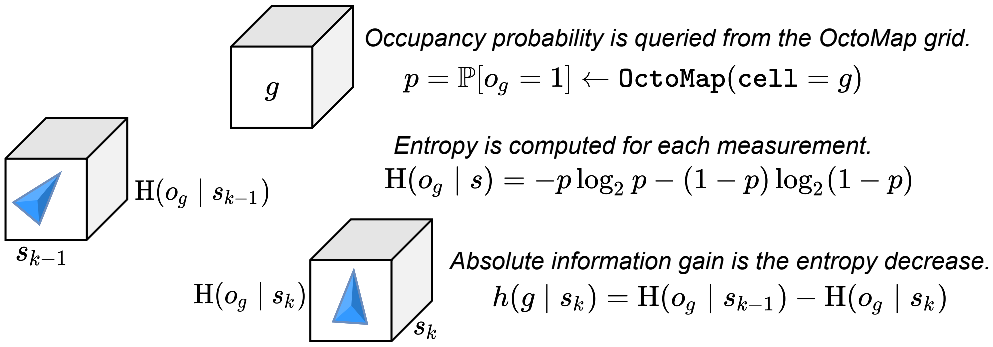

Occupancy probability is queried from the OctoMap grid.
$p = \mathbb{P}[o_g = 1] \leftarrow \texttt{OctoMap}(\texttt{cell} = g)$
g
Entropy is computed for each measurement.
$\mathrm{H}(o_g \mid s) = -p \log_2 p - (1-p) \log_2 (1-p)$
$\mathrm{H}(o_g \mid s_{k-1})$
s_{k-1}
Absolute information gain is the entropy decrease.
$h(g \mid s_k) = \mathrm{H}(o_g \mid s_{k-1}) - \mathrm{H}(o_g \mid s_k)$
$\mathrm{H}(o_g \mid s_k)$
s_k

FIGURE 4.4: The absolute information gained (about the occupancy state at a cell g) by moving from the viewpoint s_{k-1} to the viewpoint s_k corresponds to the decrease in information entropy about its occupancy state. © IEEE

Hence, the parameter λ directly affects the distance traveled within a measurement step and thus the exploration time. However, λ does not affect the NBV algorithms' coverage performance in terms of feature reconstruction. In practice, for highly constrained spaces, it is preferable that the robot explores in short bursts, thus favoring a reduced value for λ. Conversely, for highly spacious conditions, the distance constraints can be relaxed, allowing the robot to move to viewpoints further away, which in turn favors an increased value for λ. In our evaluations, the parameter is set to $\lambda = \frac{1}{2}$ to give equal weight to the gain and cost terms, as the evaluation's focus is to assess the feature reconstruction capability. With that said, the parameter λ can be readjusted to application-specific and case-specific needs, if necessary.

The gain$(\cdot)$ function is obtained by summing over the grid,

$$\mathrm{gain}(s_k) = \sum_{g \in \mathcal{G}_k} \left[h(g \mid s_k) \cdot p_\phi(g) \cdot p_v(g \mid s_k) \right] \tag{4.5}$$

Here, $h(g \mid s_k)$ is the information entropy decrease which measures absolute information gain at a cell g after placing the next viewpoint at s_k. The feature probability $p_\phi(g)$ estimates the feature membership of the cell g, *i.e.*, the chances of belonging to the drawing. The visibility probability $p_v(g \mid s_k)$ estimates the chance that the cell g is visible from s_k. The probabilities $p_\phi(g)$ and $p_v(g \mid s_k)$ respectively penalize distance from the desired feature (the drawing) and occlusion.

The entropy decrease $h(g \mid s_k)$ is obtained through the entropy function $\mathrm{H}(\cdot)$ used in information theory, as follows,

$$h(g \mid s_k) = \mathrm{H}\left(o_g \mid s_{k-1}\right) - \mathrm{H}\left(o_g \mid s_k\right) \tag{4.6}$$

The binary occupancy random variable o_g models whether the cell g is unoccupied ($o_g = 0$) or occupied ($o_g = 1$). The information entropy $\mathrm{H}(o_g \mid s_k)$ quantifies the uncertainty of the binary random variable o_g denoting the occupancy of cell g. When this information entropy decreases across two consecutive measurements, *i.e.*, when $h(g \mid s_k) > 0$, then we have "lost some uncertainty" or "gained information" about the occupancy state of the grid cell g, after the k^{th} measurement from cell s_k. This procedure is illustrated as shown in Fig 4.4.

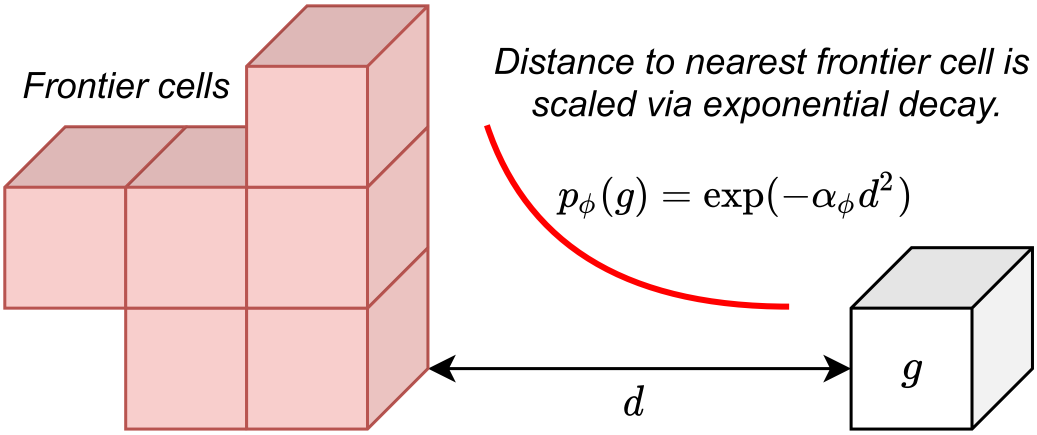

Frontier cells
Distance to nearest frontier cell is scaled via exponential decay.
$p_\phi(g) = \exp(-\alpha_\phi d^2)$
g
d

FIGURE 4.5: The feature probability approximates the probability that a grid cell g belongs to the desired feature. The frontier cells in this case represent the cells that have been determined to be part of the desired feature. © IEEE

The feature probability is modeled with exponential decay,

$$p_\phi(g) = \mathbb{P}\left[\phi_g = 1\right] = \exp\left[-\alpha_\phi \operatorname{dist}_{\mathcal{F}_k}(g)^2\right] \tag{4.7}$$

The feature indicator random variable ϕ_g models whether the cell g belongs to the drawing ($\phi_g = 1$) or otherwise ($\phi_g = 0$). The function $\operatorname{dist}_{\mathcal{F}_k}(\cdot)$ returns the shortest Euclidean distance from g to the nearest frontier cell in $\mathcal{F}_k$. The parameter $\alpha_\phi > 0$ is used to tune the exponential decay profile. Maximizing only the absolute information gain between two consecutive viewpoints would lead to seeking novel information about the state of the grid regardless of its relevance to the desired feature. Therefore, new information gained about each cell must also be scaled by the cell's probability of belonging to the desired feature. This feature probability p_ϕ is approximated using the assumption that cells near the desired feature have a higher probability of belonging to the feature. This assumption is modeled using an exponential decay profile applied onto the distance to the nearest frontier cell. We illustrate this computation as shown in Fig. 4.5.

The visibility probability at cell g from candidate s_k is,

$$p_v(g \mid s_k) = \mathbb{P}\left[v_{g,s_k} = 1\right] = \prod_{r \in \mathcal{R}_k} \mathbb{P}\left[o_r = 0\right] \tag{4.8}$$

The binary visibility random variable v_{g,s_k} models whether the cell g is visible ($v_{g,s_k} = 1$) or otherwise ($v_{g,s_k} = 0$) from the candidate s_k. An unobstructed view to the cell g from a candidate s_k must be must preferred. Raycasting is performed from s_k to g, where $\mathcal{R}_k$ contains all cells traversed by the ray. The probabilities that the ray cells are unoccluded $\mathbb{P}\left[o_r = 0\right]$ are multiplied to yield $p_v(g \mid s_k)$, *i.e.*, the probability that the cell g is visible from the candidate viewpoint s_k. The above procedure is illustrated as shown in Fig. 4.6.

During the search, all grid positions $({}^{\mathsf{b}}x, {}^{\mathsf{b}}y, {}^{\mathsf{b}}z)$ within $\mathcal{S}_k$ are attempted for scoring. For each candidate position, the orientation is searched by varying the angles α_{P} and α_{Y}. The C-NBV planner relaxes the angle α_{R} to not overly constrain the motion planner, since the normal direction about the lens has a comparably lesser effect on the information gain in the acquired image. After determining the next viewpoint s_k^*,

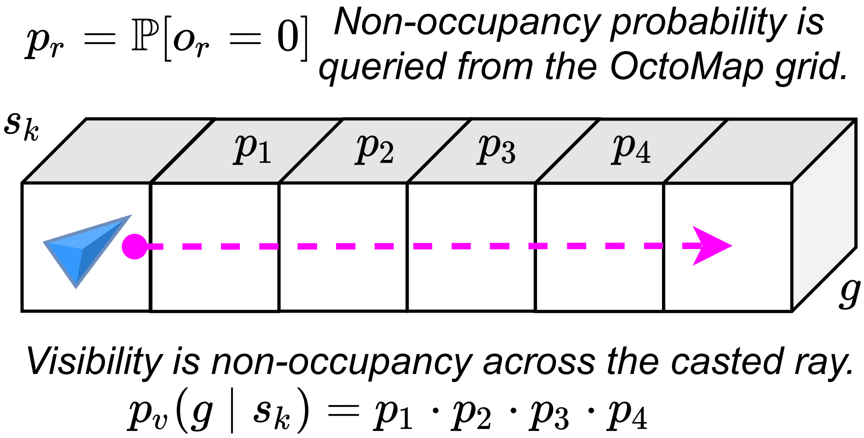

$p_r = \mathbb{P}[o_r = 0]$
Non-occupancy probability is queried from the OctoMap grid.
s_k
p_1
p_2
p_3
p_4
g
Visibility is non-occupancy across the casted ray.
$p_v(g \mid s_k) = p_1 \cdot p_2 \cdot p_3 \cdot p_4$

FIGURE 4.6: The visibility probability of a cell g from the candidate viewpoint s_k is computed as the probability p_v that all their intermediate cells are unoccupied. © IEEE

the camera's grid position (x, y, z) and orientation α_P and α_Y are set accordingly. The angle α_R is determined by the motion planner, and the camera is moved to its new pose.

With the quality metric $Q(\cdot)$ fully defined, a next viewpoint can be obtained, and the procedure elaborated above repeats until the following stop condition is met:

$$\left[\max_{s \in \mathcal{S}_k} Q\left(s\right) < \delta_Q\right] \vee \left[\operatorname{card}(\mathcal{F}_k) = 0\right] \tag{4.9}$$

That is, the routine stops once the optimal viewpoint quality falls below a certain threshold δ_Q, or when the number of frontier cells reaches zero, whichever condition is met first . This procedure is outlined in Algorithm 2.

This formulation is based on the object reconstruction algorithm developed in [64] as well as the information gain approach in [65]. However, these formulations are not well-adapted to our problem, and are either impractical (search spaces are too large and NBV selection is too slow) or infeasible (scrap objects are often too large to fully scan).

As presented above, we implement several modifications to reformulate these approaches for our requirements. We modify the frontier definition to focus on the desired feature. We also generate constrained search spaces around specific frontier cells to vastly decrease the search's time and memory costs. To determine the NBV's orientation, we constrain its range by pointing the camera at the frontier cells.

4.5.3 Guided Next Best View (G-NBV) Algorithm

This planner implements specialized modifications to the previous algorithm to radically increase performance. It does this by not only constraining the search, but also explicitly guiding it along the feature. The planner thus uses feature information to perform greedy optimizations.

Unlike the E-NVP planner, this algorithm does not assume that it explores a path. However, by guiding its search along the feature, it retains the performance advantages of path exploration. Yet, its probabilistic NBV formulation is more robust

Algorithm 2: Constrained Next Best View (C-NBV)

```
Input   : Initial view containing part of the drawing.
Output: ${}^{\mathrm{b}}D_{\mathsf{total},k_{\mathsf{stop}}}$, fully-reconstructed drawing cloud.
Initialize step, stopping condition, and viewpoint.
  step $k \leftarrow 0$
  stop $\leftarrow$ False
  ${}^{\mathrm{b}}\mathrm{T}_0 \leftarrow$ InitialViewpoint()
Search the grid until viewpoint quality is exhausted.
while stop $\equiv$ False do
    Acquire, transform, concatenate, and filter clouds.
      $({}^{\mathrm{b}}D_{\mathsf{total},k},\ {}^{\mathrm{b}}C_{\mathsf{total},k}) \leftarrow$ ProcessClouds($k$)
    Generate octree grid from cumulative clouds.
      $\mathcal{G}_k \leftarrow$ GenerateGrid($^{\mathrm{b}}C_{\mathsf{total},k},\ {}^{\mathrm{b}}D_{\mathsf{total},k}$)
    Determine and filter the frontier cells in the grid.
      $\mathcal{F}_k \leftarrow$ GetFrontier($\mathcal{G}_k$)
    Generate viewpoint search space around frontier.
      $\mathcal{S}_k \leftarrow \bigcup_{f \in \mathcal{F}_k} S(f)$
    Obtain next viewpoint with highest quality.
      ${}^{\mathrm{b}}\mathrm{T}_{k+1} \leftarrow \mathrm{argmax}_{s \in \mathcal{S}_k}\ Q\,(s)$
    Evaluate and update the stopping condition.
      stop $\leftarrow \max_{s \in \mathcal{S}_k} Q\,(s) < \delta_Q$ or $\mathrm{card}(\mathcal{F}_k) = 0$
    $k \leftarrow k + 1$
return ${}^{\mathrm{b}}D_{\mathsf{total},k_{\mathsf{stop}}}$
```

against unknown space, occlusions, and adversarial geometries. In a sense, this carries the performance advantages of the path exploration paradigm and the robustness characteristics of the probabilistic NBV formulation.

We repeat the formulation of the C-NBV algorithm until after the frontier $\mathcal{F}_k$ is determined. Despite the large performance improvements of the already reduced frontier, the search space can still become being quite large. Consider the scenario of a large object with a drawing traversing long parts of its surface. This produces a long frontier with little overlap between the individual search spaces $S(\cdot)$, causing the concatenated search space $\mathcal{S}_k$ to significantly grow.

To address this, the G-NBV planner reduces the frontier $\mathcal{F}_k$ into a single frontier cell $\hat{f}_k$. For this, the frontier cells are clustered using a suitable voxel clustering method. We use a connectedness-based clustering method, whereby any neighboring frontier cells are lumped into the same cluster. Now, the frontier is composed of several clusters, of which the nearest one (to the current pose) is chosen. The centroid $\hat{f}_k$ of this nearest cluster is computed and then used as the single-voxel frontier for generating the search space.

This offers numerous performance advantages. First, by trimming the other clusters, the planner explores any number of branches, one at a time, avoiding long and exhaustive searches. Second, since there is only one frontier cell, then the entire search space consists of a single geometric primitive around the cell, *e.g.*, a sphere around the centroid. This is a reasonable step, as frontier cells tend to surround the current reconstructed drawing's endpoints. Thus, searching around this centroid would implicitly guide the planner along the drawing.

A third advantage lies in determining orientations for the candidate viewpoints. This is normally expensive as it drastically increases the search space, since for every position there are several the camera orientations. We eliminate this problem entirely by constraining the orientation from the candidate cell s to the frontier centroid $\hat{f}_k$, thus predetermining the values for the angles α_{P} and α_{Y}. This reduces G-NBV's search to only the grid positions, since α_{P} and α_{Y} are predetermined and α_{R} is delegated to the motion planner. The search space is thus not only reduced cell-wise, *i.e.*, attempt only the search space around $\hat{f}_k$ rather than around the entire frontier, but also pose-wise, *i.e.*, search only grid positions in the search space while constraining the camera orientation.

This constraint is justified due to the localized information on the drawing. In effect, high viewpoint qualities concentrate in the unknown space around the current reconstructed drawing's endpoints. Fixing the orientation towards the centroid eliminates the exploration of alternatives where the gain is often marginal or negative. By exploiting the drawing's structure, the planner is able to rapidly select the next viewpoint.

The G-NBV routine is summarized in Algorithm 3.

Algorithm 3: Guided Next Best View (G-NBV)

```
Input  : Initial view containing part of the drawing.
Output: ${}^{\mathsf{b}}D_{\mathsf{total},k_{\mathsf{stop}}}$, fully-reconstructed drawing cloud.
Initialize step, stopping condition, and viewpoint.
  step $k \leftarrow 0$
  stop $\leftarrow$ False
  ${}^{\mathsf{b}}\mathrm{T}_0 \leftarrow$ InitialViewpoint()
Search the grid until viewpoint quality is exhausted.
while stop $\equiv$ False do
    Acquire, transform, concatenate, and filter clouds.
      $({}^{\mathsf{b}}D_{\mathsf{total},k},\ {}^{\mathsf{b}}C_{\mathsf{total},k}) \leftarrow$ ProcessClouds$(k)$
    Generate octree grid from cumulative clouds.
      $\mathcal{G}_k \leftarrow$ GenerateGrid$({}^{\mathsf{b}}C_{\mathsf{total},k},\ {}^{\mathsf{b}}D_{\mathsf{total},k})$
    Determine and filter the frontier cells in the grid.
      $\mathcal{F}_k \leftarrow$ GetFrontier$(\mathcal{G}_k)$
    Reduce the frontier set into a single frontier cell.
      $\mathcal{F}_k \leftarrow$ GetClusters$(\mathcal{F}_k,$ ClusteringMethod$)$
      $\mathcal{F}_k \leftarrow$ GetNearestCluster$(\mathcal{F}_k)$
      $\hat{f}_k \leftarrow$ GetClusterCentroid$(\mathcal{F}_k)$
    Generate viewpoint search space around centroid.
      $\mathcal{S}_k \leftarrow S(\hat{f}_k)$
    Obtain next viewpoint with orientation constraint.
      ${}^{\mathsf{b}}\mathrm{T}_{k+1} \leftarrow \underset{s \in \mathcal{S}_k}{\operatorname{argmax}}\ Q(s)\ \text{s.t.}\ {}^{\mathsf{b}}\mathrm{R}_{k+1} = \mathtt{Rot}(s, \hat{f}_k)$
    Evaluate and update the stopping condition.
      stop $\leftarrow \max_{s \in \mathcal{S}_k} Q(s) < \delta_Q$ or $\operatorname{card}(\mathcal{F}_k) = 0$
    $k \leftarrow k + 1$
return ${}^{\mathsf{b}}D_{\mathsf{total},k_{\mathsf{stop}}}$
```

4.5.4 Spatial Curve Reconstruction

By using any of the three aforementioned algorithms, we obtain the fully-reconstructed point cloud ${}^{\mathrm{b}}D_{\mathrm{total},k_{\mathrm{stop}}}$ of the drawing. This cloud is used to generate a suitable cutting path (along the drawing) that can eventually be used as a reference for cutting control. We accomplish this task of converting unstructured point clouds to ordered paths, as in the mapping (4.2), by using suitable spatial curve reconstruction methods. This is the same class of fitting methods used in the Extrapolated NVP's curve fitting step. In effect, either of the corresponding methods (NURBS-based point-distance minimization, or topological skeletonization) as explained in subsection 4.5.1 is suitable for obtaining a cutting path from the fully-reconstructed drawing. More generally, spatial curve reconstruction is researched extensively [101], [102] as is their use in fitting unordered data [103]. Such methods are of special interest to our problem within which fitting spatial curves on unordered 3-D point clouds is desired.

One common class of reconstruction methods formulate the problem as an optimization over splines—common instances include B-splines, non-uniform rational basis splines (NURBS), and Bézier curves. For example, there are iterative methods for surface fitting [104] in the presence of obstacles, as well as reconstruction [105] of self-intersecting lines. More complicated shapes have been reconstructed by partitioning them for further fitting using multiple curves. An alternative approach is using principal curves [106], [107] that are based on principal component analysis. The NURBS curves have desirable properties—they are memory-efficient and have a configurable smoothness parameter. Fitting splines to a 3-D point cloud is a non-trivial optimization problem with implementations such as in the Point Cloud Library [108].

Skeletonization algorithms instead are used to represent the connectedness of N-dimensional binary shapes and can easily represent branching paths [109]. Such methods are commonly implemented in thinning algorithms [110] for 2-D images, but extend to 3-D [111] data. Conversely, the skeletonization method identifies the set of points equidistant to at least two boundary points. In practice, this is the *medial axis* of a 2-D set of pixels, or of a 3-D set of voxels. The skeleton obtained is a voxel-wide representation of a mesh's connectedness. The 3-D skeletonization implementation as used is described in [112]. This gradually thins a point cloud (removing boundary voxels) until it yields a voxel-wide skeleton.

Each fitting approach has particular advantages and limitations for spatial curve reconstruction. One limitation of skeletonization comes from converting raw point clouds into binary voxel occupancy grids whose resolution affects the skeleton's accuracy. A finer leaf size for the voxel grid leads to better accuracy but with a robustness trade-off as sparsely-sampled point clouds can lead to fragmented occupancy (disconnected voxels) wherein the skeleton is undesirably disconnected. Conversely, splines are especially robust to noise and gaps in the point clouds wherein their functionality extends to discontinuous meshes irrespective of the voxel grid's resolution. This trade-off is illustrated in Fig. 4.7 where an increase in leaf size causes a larger average error in curve fitting for skeletonization yet negligible impact on average error for the spline-based fitting. Nevertheless, skeletonization methods can efficiently yield curves that are comparatively more robust across a wide variety of mesh geometries.

After the fully-stitched cloud is available, a spatial curve can be reconstructed using either of the aforementioned techniques. When using skeletonization, we ensure all medial axis points are fully-connected yet still accurate by choosing the smallest appropriate grid resolution; smaller branches are also pruned. Alternatively, the NURBS

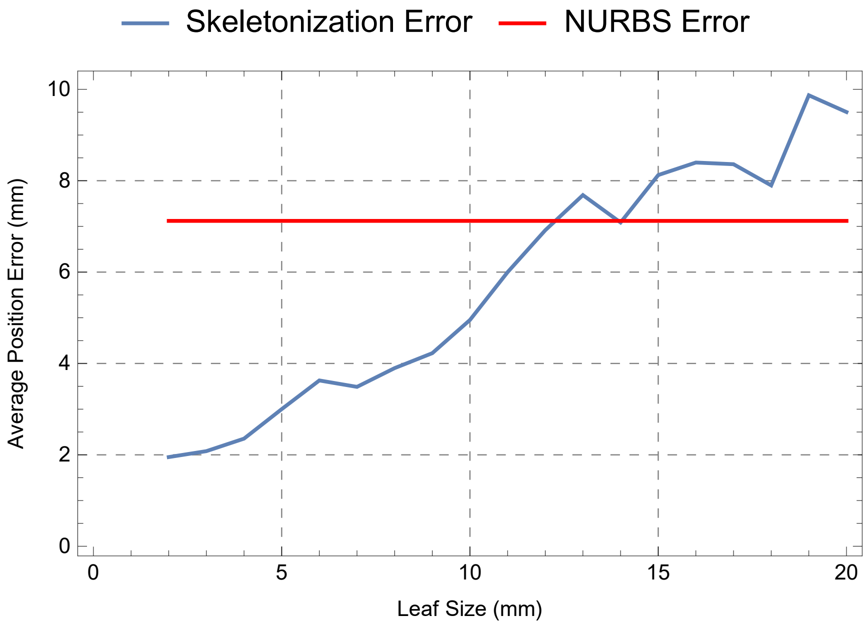


FIGURE 4.7: The impact of voxel grid resolution (leaf size) on the average skeletonization error versus average NURBS error in implementation. © IEEE

method can generate a global fit while minimizing its error by tuning the control points, degree, or smoothness constraints. In implementation, some additional preprocessing may be necessary depending on the application conditions. For instance, noisy points that do not belong to the dominant curve can be filtered using statistical outlier removal techniques. Furthermore, the point clouds may be downsampled to reduce excess accumulation of sensory data.

4.6 Simulations & Physical Experiments

This section details the approaches taken to evaluate and compare our NVP algorithms in simulation and in the physical experiments. Additionally, it defines the benchmarking metrics and ground truths used for performance quantification.

4.6.1 Design of Realistic Simulation

The simulation is configured to mimic realistic conditions and outcomes. In effect, the simulation's camera model replicates the sensing properties of the D435 stereocamera used in the experiment, *e.g.*, image size, and sensory noise modeled after the D435's noise distribution found in its datasheet [113].

For a realistic evaluation of the NVP algorithms, the test objects must carry adversarial features commonly found in the scrapyard. We select six feature categories or challenges with which we evaluate our planners. These are:

1. Smooth surface without occlusion
2. Sharp occlusion between smooth surfaces
3. Smooth occlusion along a smooth surface

4. Sharp transition between smooth surfaces
5. Highly nonsmooth surface with occlusions
6. Highly discontinuous surface

For this, we acquire high-detail 3-D scans of actual metal scrap pieces (see Table 4.1) adequately selected and retrieved from our industrial collaborator's shipbreaking yard. We add two reference objects (cylinders) to examine the robustness of each algorithm against a simple sharp self-occlusion.

4.6.2 Setup of Physical Experiment

The evaluation steps are standardized and replicated in both the simulated and experimental environments. For this, both environments are configured to reduce discrepancies between their conditions. The same red color for the drawing is used on both the simulation objects and the physical objects. Furthermore, the camera settings and the ambient lighting are kept consistent and maintained throughout the experiments. In addition, the planners are evaluated and tuned fairly. All planners share the same filter ξ, E-NVP uses $\delta_{\text{distance}} = 1$ voxel, and both NBV planners share the parameters ($\lambda = \frac{1}{2}, \alpha_{\phi} = 5$).

More importantly, the physical test objects are direct feature analogs of the simulated ones as shown in Table 4.1. This means that the NVP algorithms evaluated in either the simulations or the experiments face the same types of challenges in each feature category. For these reasons, the simulation and experimental results may be used collectively to reliably draw conclusions on the performance of the NVP algorithms.

Note that for both evaluations, the objects were fixed and evaluated one at a time. That is, for each object we first obtain a ground truth of the drawing using the method at the end of this section. Afterwards, the three NVP algorithms are run sequentially and all their results are collected in the same frame without moving the tested object. We repeat these four procedures with every object until completion.

4.6.3 Benchmarking Metrics

The following metrics are chosen to assess and quantify the efficiency and robustness of our viewpoint planners.

Number of Viewpoints

This counts the number of iterations in the exploration routine, which corresponds to the number of viewpoints visited until termination. This excludes the initial viewpoint, since it is an input to the exploration task used to generate the first round of viewpoint candidates. For instance, $n_{\text{viewpoints}} = 1$ means the robot moved to one viewpoint from the initial pose, reached the stopping condition, and then terminated the search.

This metric helps compare the number of steps taken by each NVP algorithm until termination.

Total Duration

In our evaluations, we measure only the viewpoint planning time and exclude motion planning and execution times as these are external processes. In other words, we partition the total exploration time into NVP processing time and motion-related time. Let Δt be this next view planning time which we call "Total Duration" in Tables 4.2 and 4.4. In our evaluations, this is computed as follows,

$$\Delta t = \Delta t_{\text{exploration}} - \Delta t_{\text{motion}} \tag{4.10}$$

This metric allows us to directly compare the processing duration spent by each NVP algorithm from the start of the exploration task until termination.

Average Step Duration

It is desirable to assess and compare the average processing time per iteration $\overline{\Delta t}$ taken by each NVP algorithm for a particular scenario. This is computed using the previous two metrics, as follows,

$$\overline{\Delta t} = \Delta t / n_{\text{viewpoints}} \tag{4.11}$$

This allows us to more easily perceive the trade-off between number of steps and step duration across each algorithm.

Total Displacement

We measure the spatial configuration of the chosen viewpoints independent of the robot's motion plan, by summing the displacement magnitudes between viewpoints. Specifically, let $(p_k)_{k=0}^{n_{\text{viewpoints}}}$ be the sequence of viewpoint positions, indexed in the order they were visited. This includes the initial viewpoint, since we sum the magnitudes between viewpoints. The total displacement d_{total} is,

$$d_{\text{total}} = \sum_{k=1}^{n_{\text{viewpoints}}} \|p_k - p_{k-1}\|_2 , \tag{4.12}$$

This allows us to compare the transient behavior of each algorithm, alongside the magenta polygonal chain connecting all visited viewpoints in Table 4.5.

Coverage

We use point cloud coverage as a measure of correctness for the reconstructed drawings resulting from the use of each NVP algorithm. For our evaluations, we define the coverage of a cloud ${}^{\mathsf{b}}C$ over a ground truth cloud ${}^{\mathsf{b}}G$ as,

$$\text{coverage}_G({}^{\mathsf{b}}C) = \frac{\text{card}\left[\text{vox}({}^{\mathsf{b}}C) \cap \text{vox}({}^{\mathsf{b}}G)\right]}{\text{card}\left[\text{vox}({}^{\mathsf{b}}G)\right]} \tag{4.13}$$

Here $\text{vox}(\cdot)$ voxelizes the clouds in a common and precise grid to determine their overlap, and $\text{card}(\cdot)$ is the cardinality which returns the number of voxels. The cloud ${}^{\mathsf{b}}G$ represents the ground truth to which ${}^{\mathsf{b}}C$ is compared and over which coverage is obtained. We emphasize that both clouds must be expressed with respect to a common base frame b.

In our evaluations, ${}^{b}C$ is the final reconstructed drawing ${}^{b}D_{\text{total},k_{\text{stop}}}$ and ${}^{b}G$ is accepted to represent the actual drawing in point cloud form. We discuss the method used to generate ${}^{b}G$ in the next subsection. We report coverage results in Tables 4.2 and 4.4 in two formats. The first shows the number of voxels in the ground truth cloud, while the second is a percentage.

4.6.4 Ground Truth Generation

For the purposes of coverage calculation, the ground truth cloud ${}^{b}G$ ideally represents a perfect reconstruction of the drawing in the form of a point cloud. We must obtain a good approximation for this ideal within the limitations of our stereocamera. For this, we must scan the object's local region which contains the drawing. The NVP algorithms are meant to automate this procedure of picking viewpoints. However, to improve the ground truth, we manually pick good viewpoints for the robot. The robot stays in place at each viewpoint to scan the same region repeatedly. The obtained images are averaged until the discrepancy between iterates of this average cloud falls below a preset threshold. This threshold is expressed as a proportion of the occupancy grid's voxel size. This procedure is briefly sketched in the following pseudocode.

```
define GenerateGroundTruth():
    Initialize ground truth cloud $^{b}G$
    while UserInput():
        Move the robot to the next viewpoint
        Initialize cumulative moving average (CMA)
        while Error < Threshold:
            Acquire image point cloud
            Update CMA for this viewpoint
            Update error between last two CMAs
        Concatenate latest CMA with $^{b}G$
    return $^{b}G$
```

We use this user-assisted procedure in both simulations and experiments to generate near-optimal drawing reconstructions for each test case within the sensory limitations of the stereocamera. This reconstruction has very little noise and can serve as a ground truth for coverage calculations.

4.7 Results

In this section, we present and discuss the results obtained from the evaluations in simulations and experiments.

We tabulate our results for the aforementioned benchmarking metrics in Table 4.2 for the simulations and Table 4.4 for the experiments. In addition, in Table 4.3 we visualize for each planner, some of its simulation results: viewpoint displacement, poses, and output cloud coverages. Finally, in Table 4.5 we display the experimental output of our exploration and reconstruction task (using G-NBV for planning) against each physical test object. We show the full reconstruction of the desired feature to demonstrate the effectiveness of our feature-based NVP paradigms in a real-world scenario.

We discuss the significance of our results and the nuances between our planners in the following subsections.

4.7.1 Exploration Efficiency

The planners exhibit different transient properties while exploring each object category. In general, we observe that the E-NVP average step duration is shorter—fitting and extrapolation is relatively quick—yet the planner often requires more iterations to fully explore the feature. In contrast, the NBV planners finish in less views but often take more processing time per step to search their grids.

For E-NVP, the average step durations are fairly consistent during successful runs across object categories. This means that the E-NVP total duration is largely affected by the number of steps in the exploration routine. This implies that its total duration is grows with the feature's size on the object: longer drawings on larger objects require more steps to be fully explored and thus increase total duration. This is observed for object category 1 which features the longest drawing. This sensitivity to feature size is expected, since E-NVP typically gains less information per step when compared to the NBV planners. Both the C-NBV and the G-NBV planners are less affected by feature size (meaning number of steps) and more by the total grid size (meaning average step duration) and by the scene's complexity (*e.g.*, in categories 2, 5, and 6).

The total displacements and viewpoint poses reveal that E-NVP plans in a more conservative and predictable manner and can be seen to move along the drawing. By contrast, the NBV planners tend to pick views with much larger information gain without much consideration for their positions.

In terms of total duration, the G-NBV planner is shown to be up to around ten times faster than the C-NBV, especially in large scenes requiring a larger grid. This is expected since C-NBV's frontier definition and search space generation make it much more sensitive to grid size. Even with its feature-based constraints, the C-NBV planner is the slowest in every category. This further justifies the greedy and aggressive constraints used by G-NBV to speed up exploration.

4.7.2 Robustness against Adversarial Features

Adversarial features such as occlusions, nonsmoothness, and discontinuities affect the reconstruction effectiveness of each planner differently. In the absence of adversarial features (category 1) all methods perform quite well. Even with smooth occlusions (category 3) and unoccluded sharp transitions (category 4), coverage scores are almost perfect for all planners, and the drawing is fully reconstructed.

However, we observe coverage degradation in certain scenarios. E-NVP fails to fully reconstruct the drawing against sharp occlusions (category 2). The coverage score of 86.2% on category 2 in the experimental results is not to be interpreted as high, since it missed the portion of the drawing behind the occlusion anyway, meaning this is still considered a reconstruction failure. Both NBV planners are unaffected by difficult occlusions and manage to fully reconstruct the drawing in categories 2 and 5. We also see this on the sharply occluded portion of the experiment's basket (category six), where E-NVP failed only on the occluded portion, while the NBV planners manage to overcome it and fully scan the drawing.

On that note, all planners perform quite well against surface discontinuities where voxel sizes are increased to reduce the disturbance inflicted by the gaps. The lower score of 83.9% for C-NBV in the simulation's category 6 is not due to surface discontinuities, but to grazing incidence.

This effect occurs when the ray incidence is nearly parallel to the object surface. Grazing incidence dilutes the sampling density at which point the image obtained is noisier, distorted, and poorer in information. This effect occurs in the simulation's category 6 for both NBV planners, as can be seen by the viewpoint poses and the suffering coverage near the extremities. The NBV planners may suffer suboptimal viewing angles since this optical disturbance is unmodeled by their viewpoint quality metric. E-NVP avoids this problem by maintaining an orientation quasi-normal to the surface and thus guarantees better viewing angles. Finally, we note that both NBV planners achieve similar coverage, meaning that the greedy approach of G-NBV comes at little coverage costs.

4.7.3 Planner Preference and Selection

To summarize the behavior of each planner, E-NVP extrapolates from the feature, C-NBV's search is constrained by the feature, and G-NBV's search is guided by the feature. Furthermore, each planner exhibits its own transient characteristics, strengths, and failure modes, which make it perform better or worse in particular scenarios.

If fast exploration is not a requirement, it is appropriate to use the C-NBV planner when the feature and scene are more complicated. The C-NBV planner searches more exhaustively due to its larger frontier and search spaces. Using G-NBV in exceedingly complicated scenes may yield the typical disadvantages of greedy optimization.

For simpler and smaller objects, E-NVP iterates quickly, handles gaps exceptionally well, and does not exhibit grazing incidence. Also, E-NVP works more predictably in confined spaces as its viewpoint displacement follows the drawing more conservatively, which is not guaranteed by the NBV planners.

E-NVP should be avoided when the application faces frequent occlusions and exotic surfaces. Occlusions trigger premature termination for E-NVP, as it poorly distinguishes between surface edges and drawing extremities. Meanwhile, objects with contorted surfaces may cause reachability issues with the E-NVP planner which tries to remain normal to the surface. The NBV planners overcome both issues by exploring more permissively. E-NVP is also limited to two extremities per feature. The NBV planners handle branches arbitrarily. G-NBV's frontier reduction makes it especially suited for this.

While each of the three algorithms exhibits particular strengths and weaknesses, G-NBV offers well-rounded advantages in terms of efficient scanning and effective drawing reconstruction, making it most suitable for our domain application in metal recycling. For other applications requiring feature-driven exploration, it is best to study the particularities of each problem to select a more appropriate NVP algorithm.

4.8 Discussion

In this section, we discuss specific application scenarios of the proposed methods and the associated parameters.

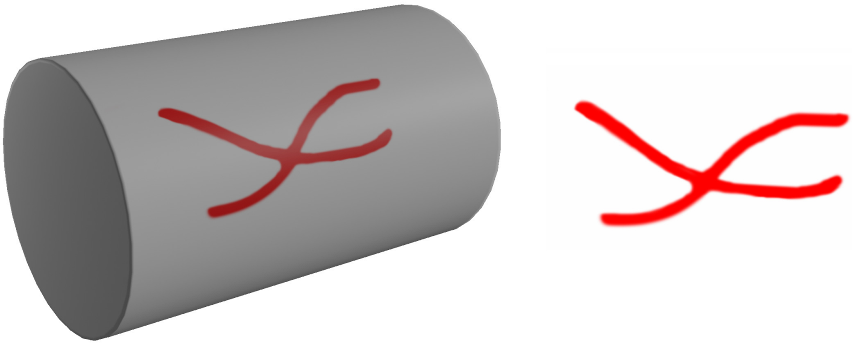


FIGURE 4.8: Test object featuring a branched drawing with four extremities. © IEEE

4.8.1 Performance against Branched Drawings

In a shipbreaking environment, the cuts are kept as simple as possible to maintain safety and control over the cutting operation and its outcomes. Although simpler cutting paths are preferred, it is still worthwhile to test against branched drawings—that is, drawings with more than two extremities.

The E-NVP method explicitly models the drawing as a curve and by design expects—and is therefore limited to—two extremities and thus cannot operate on branching curves. In contrast, the C-NBV and G-NBV methods operate at the voxel level. Accordingly, they have no conception of a branch (nor of a curve), but instead search through individual voxels in the space. This suggests that the NBV methods are capable of handling any number of branches, since as mentioned, they process the surface as voxels, regardless of branching. These algorithmic distinctions can be verified in the pseudocode listings of each of the methods in Section 4.5.

We demonstrate the above remarks in simulation on a test object with a 4-extremity branched drawing. The test object (a simple cylinder) and its branched drawing are shown in Fig. 4.8. We test each of the viewpoint planners against this test object.

The local surface reconstruction of each method is shown in Fig. 4.9. As expected, the E-NVP recovers only two extremities of the branched drawing. In contrast, both the C-NBV and the G-NBV methods, recover the entirety of the branched drawing.

4.8.2 Behavior of the NBV Algorithms near Corners

Frontier-based NBV approaches may sometimes risk trapping the viewpoints near a surface corner. The purpose of our NBV algorithms is to map the object's local surface containing the desired feature, not the full object nor its surroundings. In practice, the

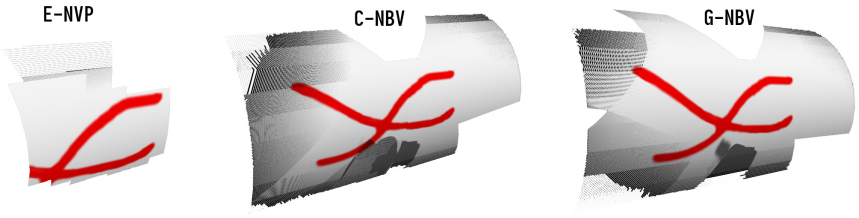


FIGURE 4.9: Local surface reconstructions around the branched drawing. © IEEE

robotic arm almost never needs to explore the object surface close enough that it risks trapping the camera in a corner. Nevertheless, we discuss how such a scenario may occur and how to avoid it. While our NBV algorithms maximize (per viewpoint) the information gain relevant to the desired feature, there is a distance tradeoff between absolute information gained and relevant feature information:

1. To maximize the information gained per viewpoint, the camera is incentivized to be placed away from the object surface. This avoids corner fixation.
2. To capture relevant feature information, the camera must remain close enough to the desired feature. This may lead to corner fixation.

Our NBV algorithms' tendency to avoid corner fixation is related to its preferred distance from the surface, which can be tuned using the decay factor α_ϕ in (4.7). By increasing α_ϕ, viewpoints closer to the surface are preferred. Conversely, lowering α_ϕ prefers more distant viewpoints. The tendency to avoid being trapped in corners can therefore be amplified or relaxed by decreasing or increasing α_ϕ, respectively. A scenario where the camera may get trapped in a corner would involve the desired featured passing through deep corners of very large objects. By sufficiently lowering α_ϕ, the planner can avoid trapping the camera in a corner.

4.9 Conclusion

This chapter develops a feature exploration and reconstruction methodology for exploring an unknown feature carrying a desired characteristic (*e.g.*, a color), located on an unknown object's surface, which lies in an unknown scene. This component serves a broader robotic system designed for automated cutting in metal scrap recycling. The component requires only an initial view containing a portion of the desired feature, and is expected to fully explore and reconstruct it. The underlying methodology is developed around next view planning to determine the next viewpoint given current scene information.

For this, we present three feature-driven next view planners that exploit the feature information collected from the scene, to rapidly and adeptly plan the next view. The first planner relies on fitting a curve to the feature from which the next view is extrapolated. The second is a probabilistic formulation on a voxel occupancy grid on which the search for the next view is constrained using feature information. The third repurposes the previous algorithm with greedy optimizations to guide the search directly via feature information. We evaluate our planners both in simulation and experiments and discuss their notable strengths and weaknesses. We note that our third planner is most suited for scrap cutting. Nevertheless, the other planners retain advantages in specific scenarios. We believe that the feature-driven paradigm for exploration as examined in this chapter can also be useful outside of scrap cutting for applications which require feature reconstruction in uncertain environments.

TABLE 4.1: Feature Categories and Test Objects for the Simulation and the Physical Experiments © IEEE

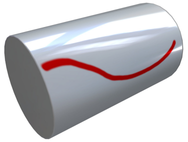
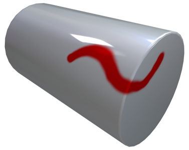
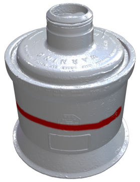
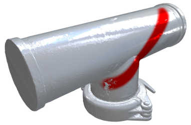
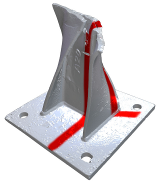
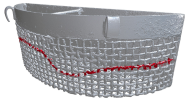
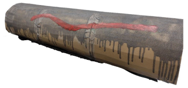
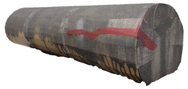
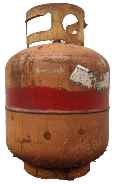
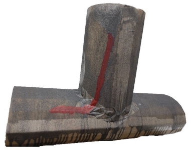
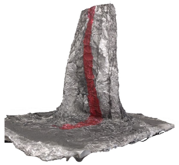
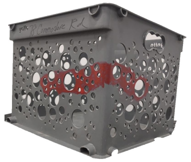
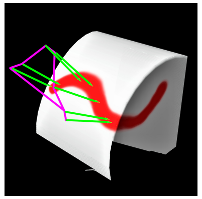
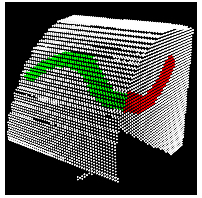
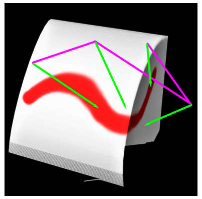
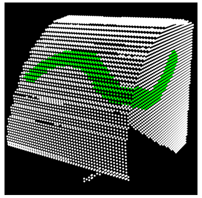
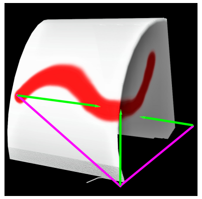
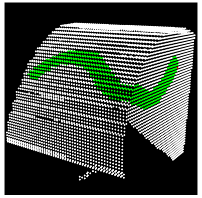
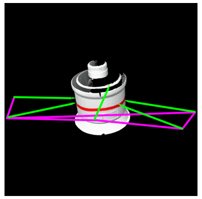
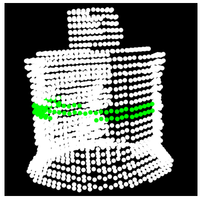
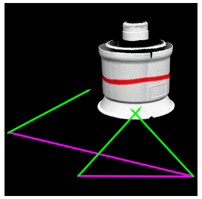
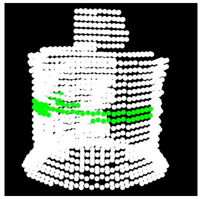
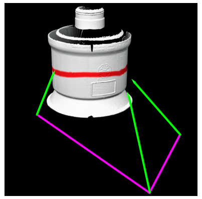
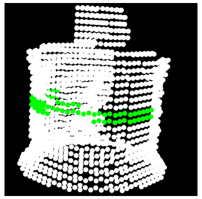
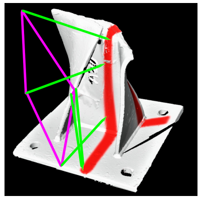
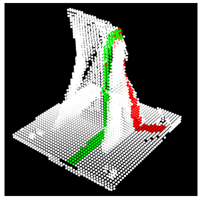
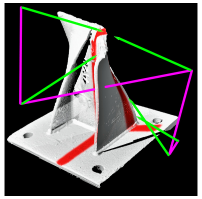
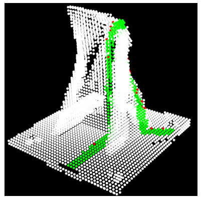
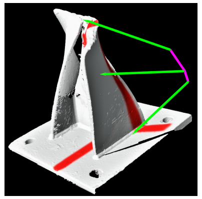
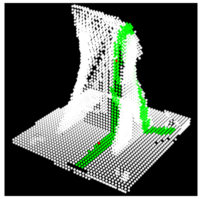
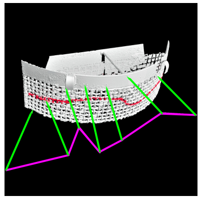
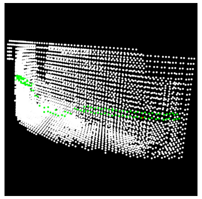
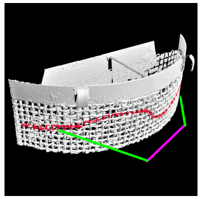
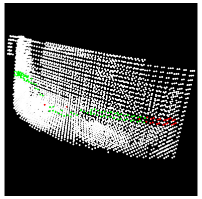
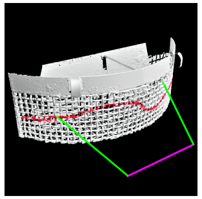
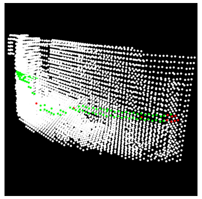
Features	*Category 1* **Smooth Cylinder** Smooth surface	*Category 2* **Sharp Cylinder** Sharp occlusion	*Category 3* **Round Tank** Smooth occlusion	*Category 4* **T-Piece** Sharp transition	*Category 5* **I-Beam** Nonsmooth surface	*Category 6* **Basket** Discontinuous surface
Simulation Objects						
Experiment Objects						

TABLE 4.2: Simulation Results: Evaluating Benchmarking Metrics for each NVP Algorithm on all Object Categories © IEEE

Challenge	**Planner**	**Viewpoints**	**Total Duration**	**Avg. Step Duration**	**Total Displacement**	**Coverage**	**Coverage**
(category)	(algorithm)	(steps)	(sec)	(sec/step)	(cm)	(voxels)	(%)
Smooth Cylinder *Category 1*	Extrapolated NVP	9	61.9	6.9	144.6	197/199	99.0
	Constrained NBV	4	808.0	202.0	48.5	197/199	99.0
	Guided NBV	4	38.6	9.6	67.8	197/199	99.0
Sharp Cylinder *Category 2*	Extrapolated NVP	8	43.4	5.4	88.3	372/569	65.4
	Constrained NBV	4	417.5	104.4	87.7	569/569	100
	Guided NBV	3	43.4	14.5	66.0	568/569	99.8
Round Tank *Category 3*	Extrapolated NVP	6	30.1	5.0	149.5	82/82	100
	Constrained NBV	3	153.3	51.1	81.8	82/82	100
	Guided NBV	3	41.6	13.9	68.1	82/82	100
T-Piece *Category 4*	Extrapolated NVP	4	18.0	4.5	81.7	179/182	98.4
	Constrained NBV	4	463.5	115.9	88.7	179/182	98.4
	Guided NBV	3	47.0	15.7	53.7	177/182	97.25
I-Beam *Category 5*	Extrapolated NVP	6	29.0	4.8	90.0	234/532	44.0
	Constrained NBV	5	549.0	109.8	108.4	525/532	98.7
	Guided NBV	3	56.4	18.8	10.5	501/532	94.2
Basket *Category 6*	Extrapolated NVP	8	46.0	5.7	94.1	153/155	98.7
	Constrained NBV	2	187.7	93.8	11.82	130/155	83.9
	Guided NBV	2	50.6	25.3	23.9	145/155	93.5

Note: The Extrapolated NVP planner backtracks to the initial point after finding the first endpoint. As a result, it records an additional step: the revisited viewpoint.

TABLE 4.3: Visualizing Viewpoints, Displacement, and Coverage for each NVP against Four Simulated Objects © IEEE

	Algorithm 1: E-NVP		*Algorithm 2: C-NBV*		*Algorithm 3: G-NBV*	
	Viewpoints	**Coverage**	**Viewpoints**	**Coverage**	**Viewpoints**	**Coverage**
Sharp Cylinder						
Round Tank						
I-Beam						
Basket						

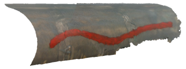
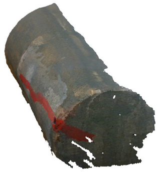
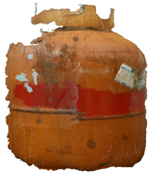
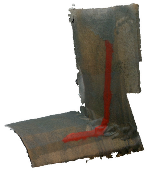
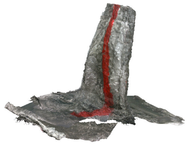
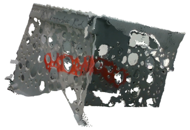
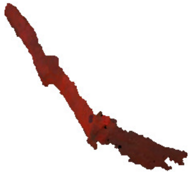
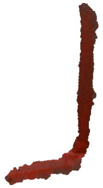
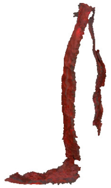
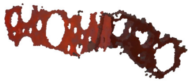
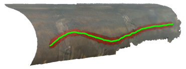
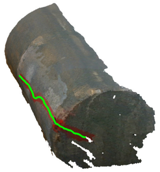
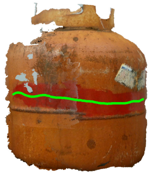
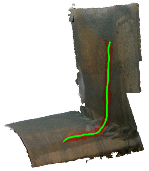
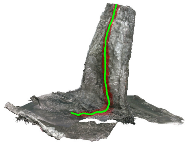
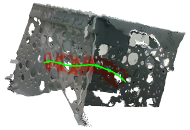
The magenta graph shows the displacement between views while the green rays show their poses.
For coverage, recovered voxels are in green while missed ones are in red.

TABLE 4.4: Experimental Results: Evaluating Benchmarking Metrics for each NVP Algorithm on all Object Categories © IEEE

Challenge (category)	**Planner** (algorithm)	**Viewpoints** (steps)	**Total Duration** (sec)	**Avg. Step Duration** (sec/step)	**Total Displacement** (cm)	**Coverage** (voxels)	**Coverage** (%)
Smooth Cylinder *Category 1*	Extrapolated NVP	8	24.3	3.0	119.5	265/270	98.1
	Constrained NBV	1	31.4	31.4	21.7	268/270	99.3
	Guided NBV	1	18.7	18.7	22.3	267/270	98.9
Sharp Cylinder *Category 2*	Extrapolated NVP	5	19.4	3.9	51.9	144/167	86.2
	Constrained NBV	3	561.2	187.1	93.7	162/167	97.0
	Guided NBV	2	68.1	34.1	62.0	165/167	98.8
Round Tank *Category 3*	Extrapolated NVP	4	13.1	3.3	67.3	324/329	98.5
	Constrained NBV	3	252.4	84.1	89.4	320/329	97.3
	Guided NBV	3	106.2	35.4	63.8	314/329	95.4
T-Piece *Category 4*	Extrapolated NVP	5	25.0	5.0	29.5	109/111	98.2
	Constrained NBV	2	178.5	89.3	44.7	110/111	99.1
	Guided NBV	1	28.4	28.4	38.7	111/111	100
I-Beam *Category 5*	Extrapolated NVP	1	9.6	9.6	28.1	92/425	21.6
	Constrained NBV	6	1015.8	169.3	122.9	410/425	96.5
	Guided NBV	7	153.3	21.9	167.7	407/425	95.8
Basket *Category 6*	Extrapolated NVP	4	6.8	1.7	32.3	117/171	68.4
	Constrained NBV	4	358.4	89.6	105.3	163/171	95.3
	Guided NBV	3	112.0	37.4	91.2	166/171	97.1

Note: The Extrapolated NVP planner backtracks to the initial point after finding the first endpoint. As a result, it records an additional step: the revisited viewpoint.

TABLE 4.5: Experimental Output: Reconstructions using the Guided NBV Planner and Corresponding Generated Cutting Paths © IEEE

Features	*Category 1* **Smooth Cylinder** Smooth surface	*Category 2* **Sharp Cylinder** Sharp occlusion	*Category 3* **Round Tank** Smooth occlusion	*Category 4* **T-Piece** Sharp transition	*Category 5* **I-Beam** Nonsmooth surface	*Category 6* **Basket** Discontinuous surface
Local Reconstruction						
Drawing Reconstruction						
Generated Cutting Path						

Chapter 5

Autonomous Oxy-fuel Cutting

As covered in Chapter 4, we develop viewpoint planning algorithms for generating the cutting path along the surface of a given input object. In this chapter, we assume that this path is available and focus on the autonomous cutting (using an oxy-fuel torch) along this path on the object surface. In sectors such as manufacturing and metal recycling, automating the industrial process of metal cutting would yield substantial improvements to productivity and safety. We propose a vision-based framework to automate metal cutting based on oxy-fuel torches, a cutting technology widely used in industry. Our framework enables a robot—equipped with a cutting torch and an eye-in-hand RGB camera—to autonomously execute a cut along a given cutting path. We achieve this in three automated tasks: vision system calibration, metal surface conditioning, and torch combustion control. The visual feedback is obtained from the camera observing the torch flame and heated region behind a tinted visor. These image frames are processed to extract meaningful features such as the heat pool's convexity and intensity. Using these features, we model the desired conditions for visual calibration, surface conditioning, and combustion control. During calibration, the vision system is configured to the current scene and setup by detecting the torch flame's centroid and its baseline intensity. Afterwards during conditioning, the metal surface is preheated to adequate conditions for initiating combustion. Finally during the cutting phase, we assume that the cutting path is given to the robot, and that it can follow the path by keeping the torch tip orthogonal to the surface. Our vision-based control algorithm automatically determines the speed of moving the torch by visually observing the metal surface's heated region and maintaining desired combustion conditions. We evaluate our framework in physical cutting experiments using a 1-DOF robot that autonomously cuts steel plates of different thicknesses. Our system successfully executes the cuts relying purely on vision without prior knowledge of the plate thicknesses.

Keywords: *Autonomous metal cutting, oxy-fuel cutting, vision-based control, calibration, surface preheating, robotic cutting.*

Notice: The work in this chapter is under review (as of July 2023) at the IEEE/ASME Transactions on Mechatronics (TMECH) and is subject to IEEE Copyright upon acceptance. An earlier version of this work is accepted at the 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) [114] and is subject to IEEE Copyright.

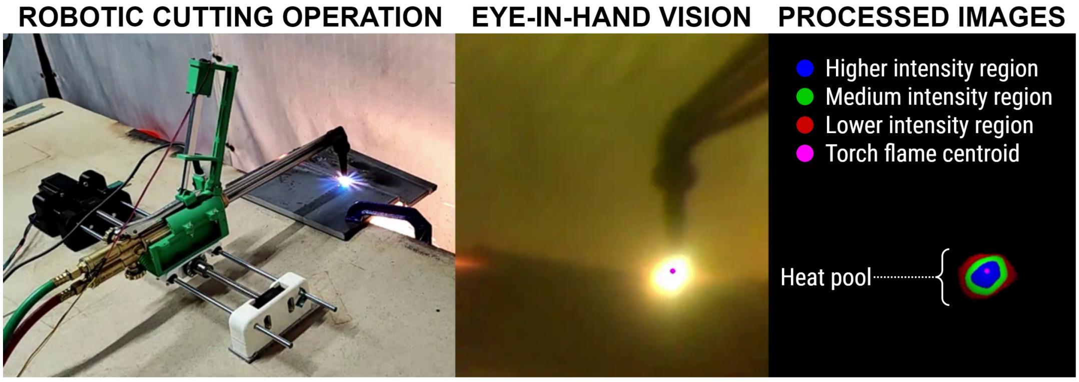


FIGURE 5.1: The proposed vision-based control algorithm enables the 1-DOF cutting robot to autonomously cut steel plates of different thicknesses. **Left:** Third-person view of the robot during control. **Center:** Eye-in-hand view of the metal surface showing the heat pool. **Right:** Processed stream of the eye-in-hand footage focusing on the shown heat pool. © IEEE

5.1 Introduction

Robotics and automation in industrial applications can offer numerous advantages such as increasing the productivity of the overall economy [115], improving the occupational safety of workers [116], and reducing the global ecological footprint [117]. In particular, industrial operations pertaining to processing metals are likely to gain importance since the demand for major metals is projected to increase significantly [45] throughout the current century. Among these is metal cutting—a key operation in sectors such as manufacturing and metal scrap recycling—whose automation can reduce operational costs [118]. For instance, this can improve the production of recycled steel, which is more efficient than processing iron ores [20] into primary steel. While there are a variety of metal cutting media (*e.g.*, laser, water-jet, plasma) each with their own advantages, limitations, and robotic setups [49], we focus on automating oxy-fuel gas cutting, which is a thermal medium based on combusting the metal surface. As mentioned in [118], oxy-fuel cutting is particulary effective against thick sections of carbon steel—and its affordability in this regard is pertinent for metal scrap cutting and recycling.

In this chapter, we formulate a vision-based framework for enabling robots (equipped with an oxy-fuel torch and an eye-in-hand RGB camera) to autonomously cut metal surfaces (Fig. 5.1) along a specified reference cutting path. This framework consists of three core tasks:

1. *Vision system calibration:* Identifying the ignited torch's flame centroid and baseline intensity, visual features explained in later sections.

2. *Metal surface conditioning:* Maintaining the lit torch over the initial position of the cutting path, preheating the metal surface for combustion.

3. *Torch combustion control:* Automatically determining the velocity of the torch along the specified cutting path to maintain the desired combustion conditions.

This vision-based approach draws inspiration from the manual techniques of skilled cutting workers, particularly those within ship-breaking yards that we surveyed. The workers visually track the location of the torch flame through a tinted visor at all times.

The process starts by placing the torch at the initial point of their desired cutting path and maintaining its position until the surface is determined to be sufficiently preheated. This is accomplished by tracking the formation and evolution of a *heat pool* (a bright blob) on the metal surface until it reaches desirable conditions for combustion (known intuitively from experience). Afterwards, the worker moves the torch along the cutting path and adjusts their speed to maintain this desired state of the heat pool until reaching the end of the path.

Along similar lines, our vision-based framework (Fig. 5.2) extracts meaningful features from the processed visual input to capture the location and properties of the torch flame and identify when the surface is sufficiently preheated for combustion. Furthermore, the heat pool's visual characteristics (shape, size, brightness, and color) are encoded using two features: the heat pool's convexity and intensity. These two features describe the heat pool's combustion state, which is tracked for updating the torch's speed along the cutting path during combustion cutting.

The main contributions of this chapter are:

- Devising a novel framework for fully-autonomous oxy-fuel metal cutting relying only on visual feedback.
- Modeling desired conditions for autonomous vision system calibration and metal surface conditioning.
- Designing a vision-based control algorithm for executing combustion cutting using an oxy-fuel torch.
- Evaluating our calibration, conditioning, and control in physical cutting experiments using a 1-DOF robot.

To the best of our knowledge, this chapter presents the first methods that automate the calibration, conditioning, and control of oxy-propane-based cutting, and therefore, provides a significant step towards autonomous metal cutting.

We emphasize that our framework formulates the torch control problem in task space and is therefore decoupled from the robotic manipulator's motion planning and control tasks. In particular, we declare the following assumptions.

1. The desired cutting path is fully-specified and known to the controller; this can be predetermined by a cutting path generation algorithm as in our prior work [55] and in Chapter 4.
2. The robot can keep the torch orthogonal to the metal surface while traversing the cutting path by adjusting its pose via conventional motion planning and control.

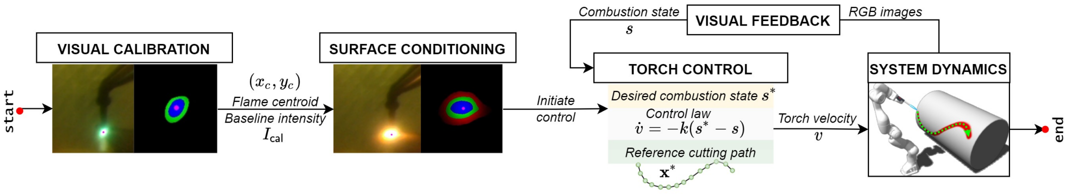


FIGURE 5.2: Overview of the proposed method. The vision system is calibrated autonomously to provide vision feedback used for controlling the torch velocity along the reference cutting path. Generating the path is addressed in our prior work [55] as well as in Chapter 4. © IEEE

Thus, given a specified cutting path, our vision-based framework enables the robot to determine the torch's cutting speed, which refers to the tangential velocity of the torch flame as it traverses the cutting path (refer to Fig. 5.3 for a visual representation). The torch flame's speed along the cutting path is crucial for successful cutting. In general, a fixed (open-loop) cutting speed does not ensure proper cutting, as is shown in our experimental results. In contrast, our experiments demonstrate that successful cuts are obtained by updating the (closed-loop) cutting speed per the metal surface's combustion state—as practiced by skilled metal cutting workers. Our experiments reveal that our framework yields successful and fully-autonomous cutting of steel plates of different thicknesses by relying purely on vision without *a priori* knowledge of the thicknesses.

5.2 Related Work

In this section, we review existing work covering the automation of cutting and welding operations, as well as visual calibration for specific tasks.

5.2.1 Automation of Cutting Operations

The automation of a particular cutting operation depends heavily on the employed cutting modality. In effect, [49] reviews a variety of cutting technologies alongside their conventional robotic setups and industrial applications. Common cutting media include machining, laser cutting, plasma cutting, oxy-fuel (gas) cutting, water-jet cutting, ultrasonic cutting. The preference for a particular cutting medium depends on domain-specific parameters, for instance, cost-effectiveness, material type, and cutting speed.

For most of these cutting media, there is a rich variety of methods to enable their automation using robots. For instance, existing methodologies covers the enabling of robots for machining [52], [119], custom-tool mechanical cutting [120], laser cutting [71], [121], plasma-arc cutting [122]–[125], hybrid induction plasma cutting [53], and contact arc cutting [126].

The automation of metal cutting operations using the oxy-fuel gas cutting modality (which this chapter addresses), by contrast, is sparsely covered in the literature. The most relevant work is in [54] wherein a reactive control architecture is developed for a gas cutting robot that trims substandard metal strips from conveyor-fed sheet metal. This control architecture employs an extensive array of sensory modalities and instrumentation tailored for strip cutting. Nevertheless, it does not cover more general cutting requirements—which this chapter addresses using a vision-based framework.

In more traditional and structured settings, CNC-based oxy-fuel cutting [127], [128] are employed extensively. However, these require the prior and complete specification of the cutting task's parameters, such as cutting speeds. In this way, automated oxy-fuel cutting is narrowed to fully-specified task parameters and predefined geometries. Nevertheless, in [118], the advantages of the oxy-fuel cutting medium are highlighted and can be further increased via improved automation. We believe that this chapter is a contribution towards this goal.

To the best of our knowledge, this is the first work on automated oxy-fuel cutting whose general-purpose formulation is independent of robot geometry. Furthermore, it relies purely on visual feedback and does not require prior specification of the metal thickness or cutting speeds.

5.2.2 Automation of Welding Operations

Welding is an operation that is related to metal cutting and existing work covering its automation extensively employs visual sensing. In [89], recent advancements in vision-based automated welding are reviewed along with particular applications such as seam tracking, deformation detection, and weld pool monitoring. In effect, vision-based sensing and control carries particular advantages such as non-invasiveness, simplicity, and inexpensiveness.

For improving welding performance, a common strategy is the processing of visual features pertaining to the welding process. Among these is the processing and tracking of weld seams. For example, in [76], welding seam features for vision-based welding control are obtained using an active vision system and extracted using image processing techniques. Weld seam features can be used for automatic multi-pass route planning [129] in metal active gas (MAG) arc robotic welding. Furthermore, for improving the seam tracking stability and precision, the vision stream can be denoised [79] using optical filters tailored to the welding process's light spectrum.

Other approaches may use additional task-specific visual features, *e.g.*, both the weld seam and the weld pool that forms on the surface. For instance, in [130], the weld seam and weld pool are extracted from images using a special-purpose Canny edge detection algorithm. These features are then used to obtain characteristic welding parameters. In addition, the weld pool's centroid can also be used as in [131] to improve the real-time tracking accuracy of the weld seam. Furthermore in [132], a vision-based method is used for seam gap measuring and seam tracking, and a controller is developed to regulate the weld pool's formation, welding current, and wire feed rate.

Weld pool features extracted from imaging are often used to improve welding performance. For example, [133] uses X-ray imaging to analyze the weld pool flow patterns by tracking particles in order to enhance its flow and the predictability of the welded component's microstructures, which affect its end-use quality. [134] uses active contours for weld pool boundary composition to overcome the interference of arc light and spatters during welding. In tungsten inert-gas (TIG) welding, [135] exploits the metal alloy's characteristics for improved image processing and pattern recognition. The weld pool's image is simplified via a weighted median filter, a statistical threshold, and a projection, after which a neural network detects the pool's edges. For real-time torch position control, [136] uses Kalman filtering on weld pool images to adjust the torch's position and improve seam tracking accuracy. [137] uses a charge-coupled device (CCD) camera's feedback filtered through specialized lenses and processed for use in control.

Weld pool analysis and control through the use imaging and image processing techniques can be employed to enhance specific aspects of the welding process—such as improved flow patterns and precise torch positioning.

The extraction of weld pool features from imaging is a common practice that can be used to optimize welding performance. For instance, in [133], X-ray imaging is employed to track particles and analyze the flow patterns of the weld pool, thus improving its flow and enhancing the predictability of the microstructures of the

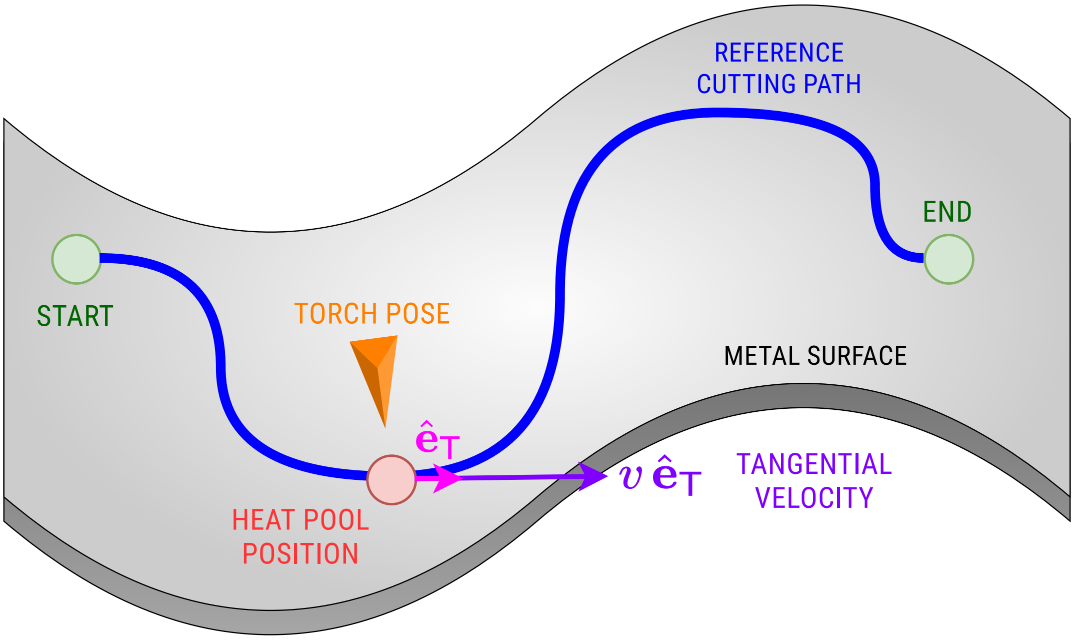


FIGURE 5.3: Illustration of the cutting control formulation showing the tangential motion along the reference path. © IEEE

welded component, which ultimately impacts its end-use quality. In [134], the interference of arc light and spatters during welding can be overcome using active contours to compose the weld pool boundary. In Tungsten Inert Gas (TIG) welding, the characteristics of the metal alloy can be leveraged [135] to facilitate and enhance image processing and pattern recognition. Specifically, a neural network model is used to detect the pool's edges whereby the weld pool's image is preprocessed using a weighted median filter, a statistical threshold, and a projection. In [136], the torch's position is adjusted and the seam tracking accuracy is enhanced during real-time torch position control by employing Kalman filtering on weld pool images. Elsewhere in [137], special-purpose lenses are utilized to filter the feedback from a CCD camera, which is then processed and used for control.

These vision-based methodologies are tailored for robotic welding and its control. As such, their specific techniques are not transferable oxy-fuel cutting and do not address the latter's particular requirements. In this chapter, we develop such a vision-based framework for the automation of oxy-fuel cutting.

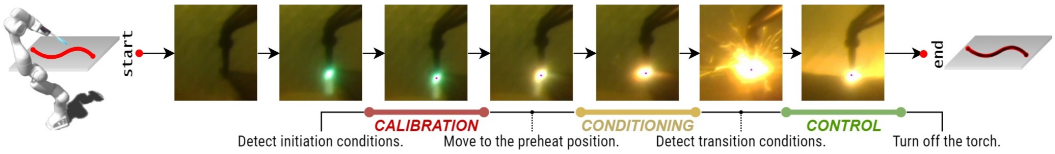

start
end
CALIBRATION
CONDITIONING
CONTROL
Detect initiation conditions.
Move to the preheat position.
Detect transition conditions.
Turn off the torch.

FIGURE 5.4: The oxy-fuel cutting operation shown as a sequence of tasks: calibration, conditioning, and control. © IEEE

5.3 Vision-based Autonomy Framework

In this section, the oxy-fuel cutting problem is formulated and our vision-based framework is delineated.

5.3.1 Cutting Problem Formulation

Referring to Fig. 5.3, the goal of autonomous oxy-fuel cutting is to move the torch's flame jet along a reference spatial path on a metal surface, such that the cut is successful and efficient. To this end, the robot must ensure that the torch flame is adequate for cutting and that the metal surface is sufficiently preheated for combustion. Cutting workers achieve these by visually tracking the flame and heat pool. In the same spirit, the robot achieves these via the tasks of visual calibration and surface conditioning. Specifically, the calibration task identifies the flame's centroid in the image frame and its baseline intensity, both of which are used downstream. After moving the torch to the initial position on the metal surface, the conditioning task monitors the formation and evolution of the heat pool until it is adequate for combustion cutting. Afterwards, the control task begins and the robot regulates the surface heat pool's combustion state while traversing the cutting path by moving the torch at an appropriate velocity.

We assume that this path is given, *i.e.*, it is predetermined by a suitable cutting path generation method. For instance, this planning task is achieved in our prior work [55] and in Chapter 4 via active vision techniques to obtain the desired cutting path and the surrounding surface normal estimates on the metal surface. Furthermore, the torch's pose is associated with two cutting parameters: the cutting angle (*i.e.*, the torch's orientation relative to the metal surface) and the standoff distance (*i.e.*, the offset between the torch tip and the metal surface). These parameters are generally kept fixed during an operation and their values are tailored to the specific needs of a particular cutting scenario. Given these parameters, a standard path planning algorithm can generate the reference path for the end-effector. In this chapter, the torch tip is assumed to be kept normal to the surface while moved along the given path and its standoff distance is maintained at 0.5 in. from the surface.

We instead focus on determining the cutting speed, *i.e.*, the velocity at which the torch is moved along this path. Moving the torch at adequate cutting speeds is essential for successful cutting. In effect, our experiments in section 6.6 provide examples (see Fig. 5.13) illustrating the outcomes of inappropriate cutting speeds on the metal surface. We emphasize that our vision-based framework does not use any *a priori* information about the plate thickness or temperature and relies purely on visual feedback (similar to what the cutting workers do). As such, the reference path is a sequence of torch tip poses to visit (which can be provided by a motion planner), wherein our novel torch controller determines the rate at which to traverse these poses, expressed as the velocity along the tangent vector (see Fig. 5.3).

The torch controller sets the acceleration along the path's tangent vector (updating the velocity) using visual feedback from the RGB camera. Expressing the control action along the local tangent vector decouples the torch control problem from the robot motion planning problem. In general settings, the tangential motion commands can be resolved into joint space via the system's inverse kinematics using conventional trajectory planners and kinematic solvers. In our experiments, however, we focus on torch control in isolation (in task space) and as such we use a 1-DOF robotic system wherein the tangent vector coincides with the robot's motion.

5.3.2 Vision-based Scheme for Oxy-fuel Cutting

To address the aforementioned oxy-fuel cutting problem, our framework leverages visual feedback. This framework is summarized in Fig. 5.2 and its tasks are sequenced in Fig. 5.4. Here we present an overview of the three main tasks: Calibration, Conditioning, and Control. We will then present their implementations in Sections 5.4, 5.5 and 5.6.

Vision System Calibration

After ignition, detect the torch flame and validate the conditions necessary for initiating the calibration routine to ensure that the vision system's calibration is reliably and satisfyingly completed. Our vision-based system monitors the flame features and determines the calibration state, preventing the system to prematurely engage, *e.g.*, when the flame fluctuates or when there is excess noise in the image frame. It also reports in real time which aspects are preventing the calibration stage to finalize. Upon successful calibration, the system records the torch flame's centroid and its baseline intensity (defined in Section 5.4.3) in the image frame to be used in subsequent (downstream) computations.

Metal Surface Conditioning

Following calibration, the torch is moved to the preheat position (the initial position on the cutting path) wherein the torch flame begins to heat the metal surface. As the heat pool forms and evolves, its features are monitored until the transition conditions are all determined to be reached. Upon successful conditioning, the surface is sufficiently hot for transitioning to the combustion phase and thereby the torch control task.

Vision-based Torch Control

With the surface sufficiently heated, the robot transitions into the combustion phase by engaging the oxygen bypass lever. In doing so, the robot enters the torch control loop and traverses the cutting path by regulating the velocity to maintain the desired pool combustion. Specifically, the heat pool's combustion state is obtained from visual feedback and its desired value is tracked by adjusting the torch's velocity. Upon reaching the end of the cutting path, the cutting operation is completed and the flame is turned off.

5.4 Visual Feature Extraction

To obtain visual feedback throughout the tasks of the vision-based framework, visual features pertaining to those task must be extracted. The visual characteristics of the torch flame and the heat pool are of particular interest, such as their size, shape, brightness, and color. Specifically, the camera images are processed (simplified and filtered) after which the features are extracted. This section develops two primary features: heat pool convexity (related to shape) and heat pool intensity (related to size, brightness, and color). From these, other task-specific features can be derived. For instance, during torch control, these two pool features are computed and combined to describe the pool's combustion state s.

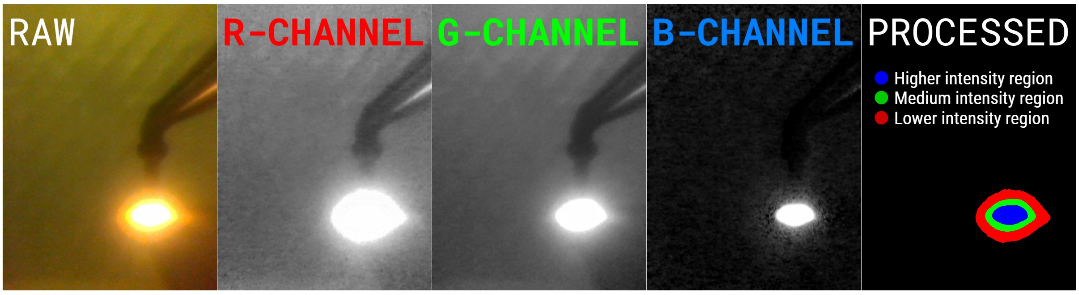


FIGURE 5.5: The raw RGB image from the camera, its red channel, green channel, blue channel, and processed version. © IEEE

5.4.1 Image Processing

The image is simplified such that the heat pool's features are emphasized. The features of interest are visual surrogates for shape and temperature. In the heat pool, the color distribution of light emissions can be used to estimate a relative temperature profile. This is because visible light of higher frequencies is emitted at higher energy states. While color-based temperature estimations are found with varying sophistication [80], [138], [139], we adopt the heuristics that follow.

We restrict the RGB color space to four discrete colors listed in order of decreasing temperature: blue, green, red, and black. In this discretized color space, blue light emissions correlate with the highest temperatures (higher electromagnetic frequency), green light with moderate temperatures (intermediate frequency), and red light with the lowest temperatures (lower frequency). Black denotes low light emissions and thus negligible temperatures. This phenomenon is confirmed in the heat pool footage. Consider in Fig. 5.5 the raw camera image and its color channels. The highest heat intensity is nearest to the flame center, and decays with distance. In effect, the blue channel's highest brightness is nearest to the pool center, whereas the green and red channels include moderate and lower temperature portions of the pool.

With this, the raw image is processed as follows:

1. Binary-threshold each channel using high cutoffs.
2. Color each channel's non-zero pixels (red, green, blue).
3. Eliminate channel overlap via `XOR` operations.
4. Merge the channels, yielding a segmented heat pool.

These are specified more precisely in the procedure `ProcessImage` shown below.

```
define ProcessImage(image $\mathbf{M}$):
    $(\theta_R, \theta_G, \theta_B) \leftarrow (240, 240, 240)$
    for each channel $k \in \{R, G, B\}$ do
        for each pixel $(i, j) \in \mathbf{M}_k$ do
            if $\mathbf{M}_k[i, j] < \theta_k$ then $\mathbf{M}_k[i, j] \leftarrow 0$
            else $\mathbf{M}_k[i, j] \leftarrow 255$
    $\mathbf{M}_R \leftarrow \mathbf{M}_R \oplus \mathbf{M}_G$  (XOR operation)
    $\mathbf{M}_G \leftarrow \mathbf{M}_G \oplus \mathbf{M}_B$  (XOR operation)
    return $\mathbf{M}$
```

This simple and efficient channel-wise thresholding procedure emphasizes the pool's geometric and thermal information. The shape and size of the pool is preserved and its color information is quantized as the image contains only four colors. This enables efficient representation and tractable reasoning about the heat pool, its features, and its combustion state. In addition, the thresholding eliminates unnecessary features from the image and effectively reduces background noise. Further noise rejection steps are elaborated in Section 5.4.3.

5.4.2 Heat Pool Convexity

Utilizing the processed image, we design a measure of the pool's shape using its convexity. During cutting, when the pool's shape is relatively more convex, then the surface conditions are more adequate for combustion. Conversely, when the pool's shape is more concave or exhibits significant convexity defects (see Fig. 5.6), then the surface conditions are more adverse for combustion. This is since a higher relative convexity is indicative of heat that is more concentrated in the heat pool. While there are sophisticated approaches to quantify a closed contour's convexity [140], we compute the area ratio between the pool contour against its convex hull.

Formally, let K be the blue (innermost) contour in the heat pool in the image; this is a set of pixel coordinates of the simple closed curve. We define the pool convexity as $c = |K|/|\operatorname{conv} K|$ where $|K|$ denotes the area of the blue contour and $|\operatorname{conv} K|$ the area of its convex hull. This yields the following desirable properties: (1) $c \in (0, 1]$ where $c = 1$ is a perfectly convex shape. (2) A higher value of c denotes a higher convexity and thus correlates with a higher combustion state. (3) The ratio c is invariant to pool's size or color, it only quantifies its shape. More precisely, it is dimension-invariant, pose-invariant, and color-invariant. In practice, restricting the contour K to the hottest layer in the pool, *i.e.*, the pool's blue channel, yields more robust convexity values. Moreover, K is retrieved computationally as the largest contour closest to the torch flame centroid (obtained previously from the calibration step).

5.4.3 Heat Pool Intensity

The pool intensity is designed to convey, from the processed image, the pool's color and size. The processed image $\mathbf{M}$ (in array form) contains pixels p considered tuples (x_p, y_p, c_p) of the pixel coordinates and color. The design and computation of the pool intensity i are delineated below.

1. Weigh all pixels p based on color (black, red, green, blue).

This captures size (total non-zero pixels) and color (via weights) to approximate the relative temperature differences in the quantized color space. The weight $w(p)$ assigned at a pixel p depends on its color c_p which can be black, red, green, or blue.

$$w(p) = \begin{cases} 0, & \text{if } c_p = \text{“Black”} \\ w_{\mathsf{R}}, & \text{if } c_p = \text{“Red”} \\ w_{\mathsf{G}}, & \text{if } c_p = \text{“Green”} \\ w_{\mathsf{B}}, & \text{if } c_p = \text{“Blue”} \end{cases} \tag{5.1}$$

The weights $w_{\mathsf{R}} < w_{\mathsf{G}} < w_{\mathsf{B}}$ are constants. We recall that blue pixels are significantly hotter than green pixels, which are hotter than red pixels; and black pixel are

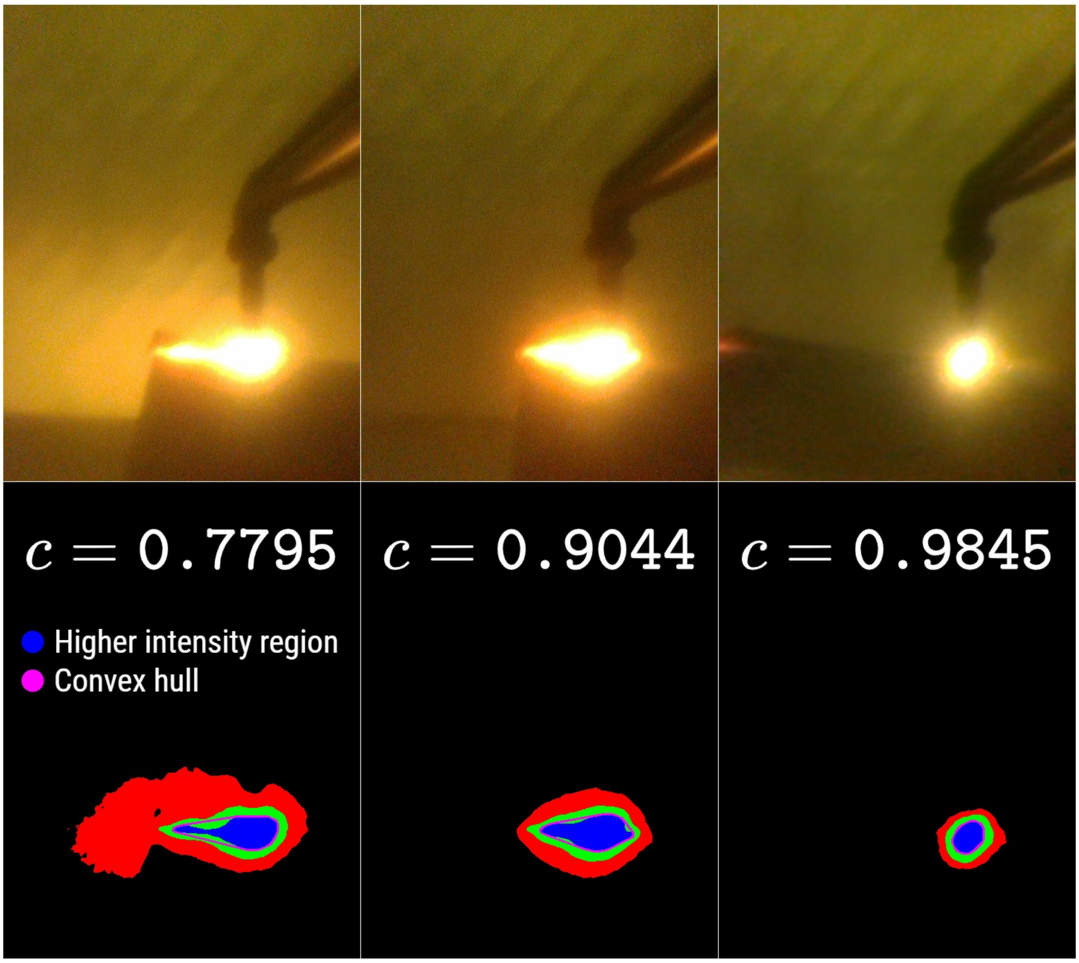


FIGURE 5.6: Demonstrating the pool convexity value (in white) along with the blue layer's convex hull (in magenta). **Left:** The pool has significant convexity defects (deviations from its convex hull), reflected in its low pool convexity of 0.7795. **Center:** The pool has moderate convexity defects and has a pool convexity of 0.9044. **Right:** The pool has negligible convexity defects and has a high pool convexity of 0.9845. © IEEE

considered to have negligible heat. In our case, $(w_\mathsf{R}, w_\mathsf{G}, w_\mathsf{B}) = (0.01, 0.04, 0.16)$ to approximate the nonlinear temperature differences between colors.

2. Weigh all pixels p via a radial decay function $g(p)$ centered at the torch flame's centroid (x_c, y_c), obtained from calibration.

This approximates the nonlinear radial decay effects of heat transfer, *i.e.*, the heat at a pixel decays quickly with distance from the torch flame's centroid. Moreover, this radial decay provides robustness against noise (Fig. 5.7) and undesirable effects such as: (1) light pollution, (2) sudden sparks, slag, or streaks, and (3) residual heated regions away from the heat pool. Here, a bivariate Gaussian function is appropriate as it models radial exponential decay using simple parameters. We observe the effect of weighing via Gaussian decay in Fig. 5.7. The 2D Gaussian function $g(p)$ assigns a decay factor to each pixel based on its distance from the torch flame centroid (x_c, y_c),

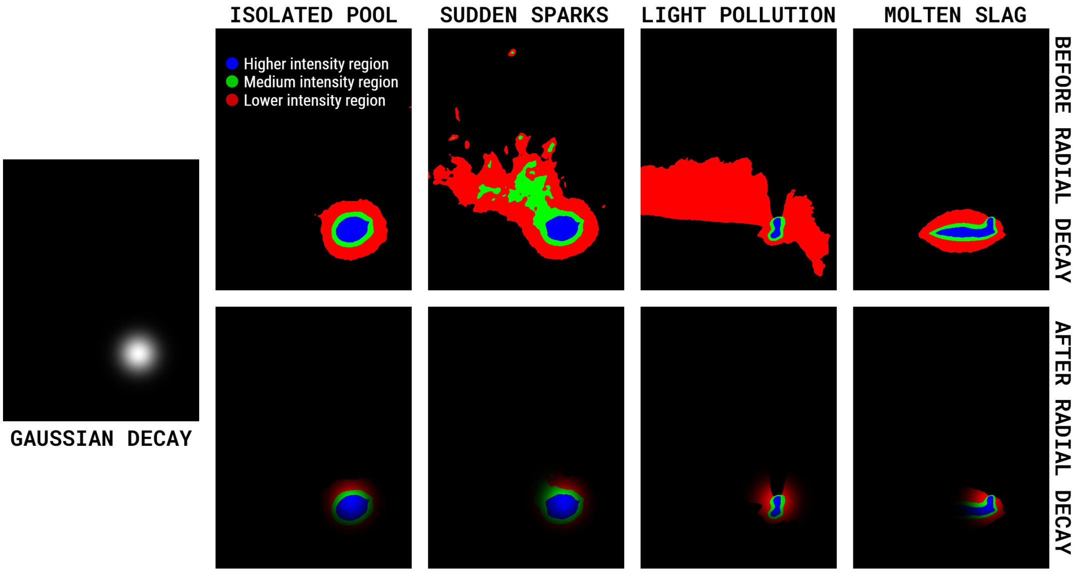


FIGURE 5.7: The Gaussian radial decay (leftmost image) is centered at the torch flame's centroid. In the various scenarios shown (top images), the decay reduces unwanted effects and noise (bottom images), as well as scales the intensities based on the distance from the torch flame's centroid. © IEEE

which is determined at calibration. This is defined as:

$$g(p) = \exp\left[-\frac{(x_p - x_c)^2}{2\sigma_X^2} - \frac{(y_p - y_c)^2}{2\sigma_Y^2}\right] \tag{5.2}$$

Here, σ_X and σ_Y determine the radial decay rate in each axis. In our case, $\sigma_X = \sigma_Y = 30\,\text{px}$ (see the Gaussian in Fig. 5.7).

3. Sum all weighted pixels p into an unscaled intensity I.

Using the aforementioned definitions, an absolute sum yields the unscaled intensity defined as $I = \sum_{p \in \mathbf{M}} g(p)w(p)$. At calibration, we average the unscaled intensity I over a time window (see Section 5.5.1) to obtain a baseline intensity value I_{cal} against which all subsequent intensities can be scaled.

4. Scale the sum I by the baseline I_{cal}, apply a saturation cutoff $\tilde{I}_{\text{sat}}$, and normalize to yield the pool intensity i.

Let $\tilde{I} = I/I_{\text{cal}}$ be the intensity expressed relative to the baseline I_{cal}. This expresses the intensities with a physical meaning, for instance $\tilde{I} = 1$ is an intensity equal to that of the torch flame, while $\tilde{I} = 5$ is an intensity five times greater. This simplifies the interpretation of intensity values and the tuning of desired intensity values. To design the pool combustion state, the pool intensity must be normalized for combination with the pool convexity. For this, a saturation intensity $\tilde{I}_{\text{sat}}$ is set to be greater than the maximum values of $\tilde{I}$ observed during trial cutting runs. In our experiments $\tilde{I}_{\text{sat}} = 10$, *i.e.*, intensity saturates at 10 times the baseline. We thus normalize the pool intensity as $i = \min(\tilde{I}_{\text{sat}}, \tilde{I})/\tilde{I}_{\text{sat}}$ with $i \in (0, 1]$.

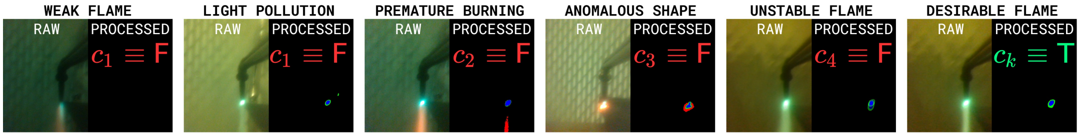


FIGURE 5.8: During visual calibration, the formulated predicates in Section 5.5.1 can detect a variety of anomalies, some of which are shown. For example, the anomalies of a weak flame or background light pollution can be detected when the c_1 predicate is false. The image shown for the unstable flame anomaly overlays two consecutive frames, showing the flame's flickering detected via a high gradient ($c_4 \equiv$ F). The desirable flame scenario is achieved when all predicates are true ($c_k \equiv$ T). Here, $c_k = C_{\mathsf{cal},k}$. © IEEE

5.4.4 Heat Pool Combustion State

As the purpose of the controller is to maintain a sufficient combustion for cutting, the visual feedback must be designed to describe the heat pool's combustion state. We note that excessive combustion indicates an overly slow torch velocity, while deficient combustion indicates an overly fast velocity, and an adequate velocity leads to a desired combustion state. The extracted visual features must then reflect this negative relationship between cutting speed and combustion state. As such, the visual feedack provided to the controller is a heat pool combustion state descriptor s computed from the camera's RGB feed.

Finally, the pool combustion state descriptor is defined as $s = \lambda c + (1 - \lambda)i$ where the relative emphasis on pool convexity or pool intensity is tuned with $\lambda \in [0, 1]$. This yields a desirable normalized state descriptor $s \in (0, 1]$. In our experiments, we set $\lambda = \frac{1}{2}$. This descriptor, by design, captures the positive relationship between convexity and combustion state (whereby a higher convexity indicates more concentrated combustion), and the positive relationship between intensity and combustion state (whereby a higher intensity indicates a stronger combustion). These modeling choices are heuristic-based yet provide a sufficiently accurate and efficient visual feedback signal for the control in Section 5.6.

5.5 Visual Calibration & Surface Conditioning

In this section, we cover the visual calibration and surface conditioning procedures as well as their predicates for initiation and transition to combustion, respectively.

5.5.1 Vision System Calibration

During autonomous calibration, to ensure an acceptable input on which the vision system is calibrated, we require a particular set of predicates, *i.e.*, Boolean variables $C_{\mathsf{cal},k}$, to be true. These initiation conditions in the image input, necessary for starting the calibration procedure, are specified below.

1. Single contour

A proper torch flame is concentrated in a single blob. Conversely, a multiplicity of blobs in the processed image indicates noise, light pollution, or other undesriable effects. As such, the set of contours $\mathcal{C}_{\mathbf{M}}$ in the processed image $\mathbf{M}$ must be a singleton. This is expressed as $C_{\mathsf{cal},1}(t) \equiv (\text{card } \mathcal{C}_{\mathbf{M}} = 1)$ where card is the cardinality.

2. Green-blue color profile

A proper torch flame must only consist of green and blue colors in its RGB channels after thresholding. In actuality, when the torch flame has red content in its color, this generally indicates impurities which can be due to uncleaned torch tip, faulty equipment, excess propane flow, or excess proximity to the metal surface. This condition is expressed as $C_{\mathsf{cal},2}(t) \equiv (\mathbf{M}_{\mathsf{R}} = \mathbf{0})$ wherein the red color channel of the processed image contains only zeroes.

3. High convexity

The torch flame must be sufficiently convex to ensure a focused and concentrated flame adequate for calibration (and cutting). This is expressed as the Boolean variable $C_{\mathsf{cal},3}(t) \equiv (c(t) > \theta_c)$ where the convexity value $c(t)$ is compared against the preconfigured threshold θ_c (in our experiments, $\theta_c = 0.95$).

4. Low intensity gradient

The torch flame should be sufficiently stable during calibration and must not flicker or fluctuate. This is expressed as the predicate $C_{\mathsf{cal},4}(t) \equiv (\frac{\mathrm{d}}{\mathrm{d}t}i(t) < \delta_i)$ where the torch flame's intensity gradient $\frac{\mathrm{d}}{\mathrm{d}t}i(t)$ is approximated using finite differences and the threshold δ_i is preconfigured (in our experiments, $\delta_i = 0.2$).

Finally, these predicates must simultaneously true for an adequate calibration of the vision system. The Boolean function $C_{\mathsf{cal}}(t) \equiv \bigwedge_k C_{\mathsf{cal},k}(t)$ is a conjunction of these predicates and, when true, indicates acceptable calibration conditions.

When $C_{\mathsf{cal}}(t)$ is true throughout a time window T_{cal} (preconfigured), the calibration procedure is initiated. Throughout T_{cal}, the centroids of the blue channel $\mathbf{M}_{\mathsf{B}}$ (indicating the blue region in the flame) are collected and averaged into the calibrated flame centroid (x_c, y_c). Similarly, the absolute intensity $I(t)$ (computed from the processed image $\mathbf{M}$) is collected and averaged into the baseline intensity I_{cal}. Additionally, the torch flame's size can be recorded as the average number of nonzero pixels N_{cal} in the processed image. In our experiments, the time window is $T_{\mathsf{cal}} = 50$ frames.

If any of the predicates fail during calibration (Fig. 5.8), then the calibration is halted and the reason is specified for troubleshooting. In doing so, the autonomous operation never proceeds without an adequate calibration of the torch flame.

5.5.2 Metal Surface Conditioning

After calibration, the torch is then moved to the preheat position to begin heating the surface and forming the heat pool. Since the torch flame centroid obtained, the location of the heat pool is now known to the vision system. The heat pool is monitored as it evolves to detect when it is adequate to transition to the combustion phase and torch control task—essentially signaling that the preheating is complete. In the same manner as the calibration step, this adequacy is described by the following predicates.

1. Concentric colors

A proper heat pool must have concentric colored regions in the processed image. That is, the red layer contains the green layer, which in turn contains the blue layer. This

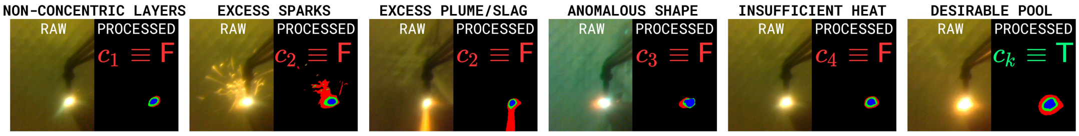


FIGURE 5.9: During surface conditioning, the predicates in Section 5.5.2 can detect a variety of anomalies, some of which are shown. For example, the anomalies of excess sparks or excess slag can be detected when the c_2 predicate is false. The desirable flame scenario is achieved when all predicates are true ($c_k \equiv \mathsf{T}$). Here, $c_k = C_{\mathsf{con},k}$. © IEEE

recursive inclusion can be expressed as the predicate $C_{\mathsf{con},1}(t) \equiv (\mathcal{N}_{\mathsf{B}} \subset \mathcal{N}_{\mathsf{G}} \subset \mathcal{N}_{\mathsf{R}})$ where $\mathcal{N}_k$ is the set of nonzero pixels in the channel k. This detects distortions or anomalies in the pool that must be corrected before control.

2. Red color proportion

An adequately heated surface yields a heat pool with a sufficient (but not excessive) proportion of red color. This indicates that the surface surrounding the torch flame is desirably heated. This requirement can be expressed as the predicate $C_{\mathsf{con},2}(t) \equiv (\frac{\operatorname{card} \mathcal{N}_{\mathsf{R}}}{\operatorname{card} \mathcal{N}} \in [\theta_{\mathsf{RL}}, \theta_{\mathsf{RU}}])$ where the thresholds $\theta_{\mathsf{RL}}, \theta_{\mathsf{RU}}$ are preconfigured (in our experiments, $[\theta_{\mathsf{RL}}, \theta_{\mathsf{RU}}] = [0.45, 0.70]$) and the set $\mathcal{N} = \mathcal{N}_{\mathsf{R}} \cup \mathcal{N}_{\mathsf{G}} \cup \mathcal{N}_{\mathsf{B}}$.

3. High convexity

The heat pool must be sufficiently convex to ensure concentrated heat for adequate combustion cutting. This is expressed as $C_{\mathsf{con},3}(t) \equiv (c(t) > \theta_c)$ where the convexity value $c(t)$ is compared against the preconfigured threshold θ_c (in our experiments, $\theta_c = 0.95$).

4. Heat pool size

The heat pool on the metal surface must be large enough, indicating that a sufficiently wide area on the surface is hot enough to sustain combustion cutting. This can be measured relative to the torch flame size N_{cal}, thus giving it a physical meaning. This predicate is expressed as $C_{\mathsf{con},4}(t) \equiv (\frac{\operatorname{card} \mathcal{N}}{N_{\mathsf{cal}}} > \theta_N)$ where θ_N is the preconfigured size threshold (in our experiments, $\theta_N = 8.5$).

All predicates $C_{\mathsf{con},k}$ must be true to indicate surface conditions that are adequate for transitioning to the combustion phase and the torch control task. The Boolean function $C_{\mathsf{con}}(t) \equiv \bigwedge_k C_{\mathsf{con},k}(t)$ is a conjunction of these predicates and, when true, indicates acceptable transition conditions. When $C_{\mathsf{con}}(t)$ is true throughout a time window T_{con} (preconfigured), then the combustion phase is started and the robot enters the torch control loop. In our experiments, the time window is $T_{\mathsf{con}} = 1$ sec.

If any of the predicates fail during surface conditioning (Fig. 5.9), the reason is specified for troubleshooting, and the controller does not start without adequate conditioning.

5.6 Vision-based Torch Control

The combustion control task assumes that calibration and conditioning are completed, meaning that the torch flame's centroid and baseline intensity are determined (so that

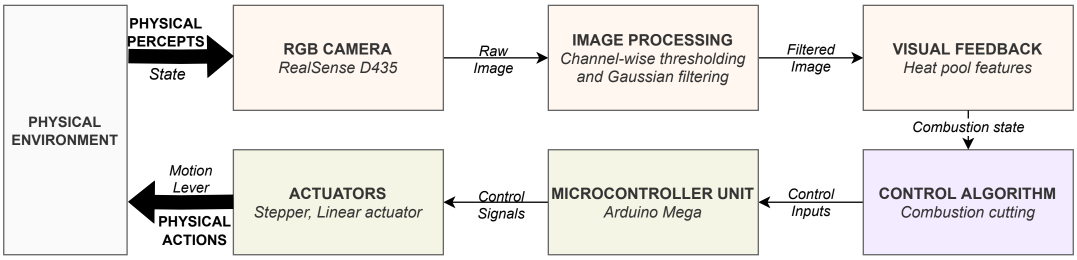


FIGURE 5.10: Block representation of the 1-DOF robotic system, its components, and its perception and control pipelines.

combustion state s is computable) and that the metal surface is sufficiently preheated for combustion to take place.

5.6.1 Modeling Assertions

The controller is designed under the following assertions:

The pose transform between the torch flame and camera is fixed.

The camera is rigidly attached to the torch, observing its tip from a fixed viewpoint throughout cutting and keeping the torch flame stationary in the image frame. After calibration, the torch flame's centroid in the image frame does not change.

The pool combustion state and torch speed have a negative relationship.

The pool combustion state is highly-correlated with the pool's temperature. A faster torch speed reduces the amount of heat transferred to the local metal surface, thereby yielding a lower pool temperature and therefore a lower pool combustion state. This is also confirmed empirically using a cutting torch and metal plates. The pool combustion state drops at higher torch speeds, and increases at lower ones. Note that the combustion state is taken to be an instantaneous measure of the most immediate pool, and does not concern residual heated regions elsewhere on the surface.

There exists a desired speed at which the desired combustion state is maintained.

This results from the negative relationship between combustion state and torch speed. Moving the torch too fast produces a deficient pool combustion, yielding poor or no cuts. Conversely, moving too slow produces an excessive combustion, yielding inefficient and badly textured cuts. There is a range of intermediate speed values yielding adequate combustion among which exists a desired pair for speed and combustion state.

5.6.2 System Variables

We formally define the variables for heat source's tangential velocity and the heat pool combustion state as follows:

- $v(t) \in \mathbb{R}_{\geq 0}$ is the tangential velocity of the heat source (torch flame centroid) on the reference cutting path.
- $s(t) = \phi(v(t)) \in \mathbb{R}_{>0}$ is the pool combustion state in the image frame, where ϕ models the map from v to s.

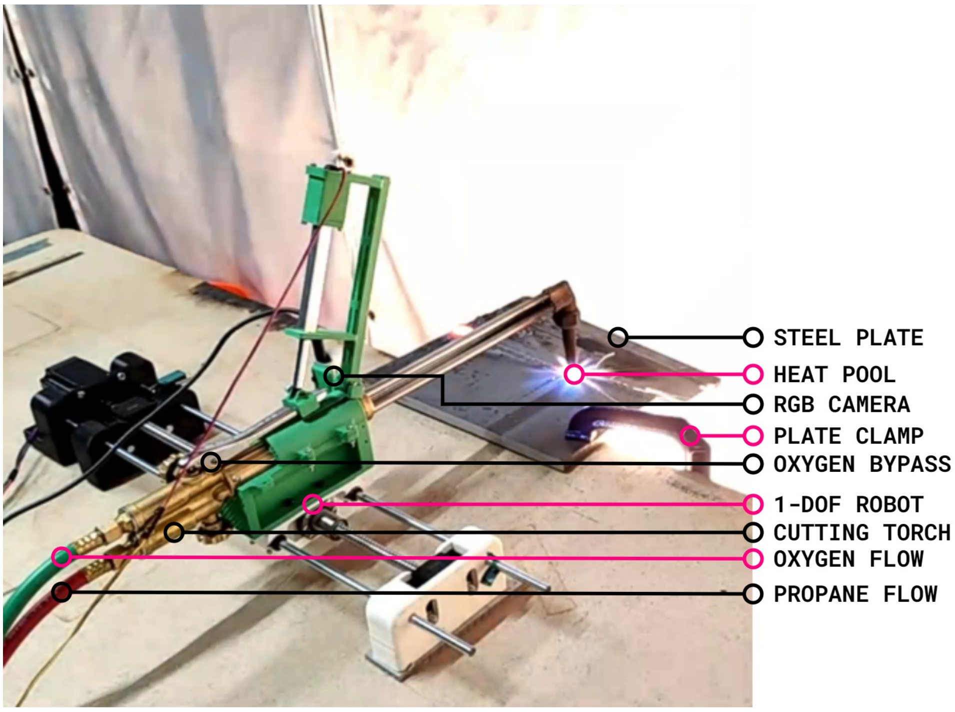


FIGURE 5.11: Steel plates of varying thicknesses are clamped and secured before cutting. The 1-DOF robot equipped with its camera and cutting torch is installed, connected to oxy-propane sources and to its computer communications. The system can then receive feedback from the camera to perform the aforementioned vision-based control and cut the plate. © IEEE

- s^* is the desired state and v^* is the desired velocity.

The function $\phi : v \mapsto s$ maps from the torch velocity to the combustion state and thus admits the following conditions:

1. ϕ is strictly positive: $\phi(v) > 0, \forall v \in \mathbb{R}_{\geq 0}$. This is since $s > 0$ due to $c, i > 0$ as defined in Section 5.4.4.
2. ϕ is monotonically decreasing with respect to v: $\frac{\mathrm{d}}{\mathrm{d}v}\phi(v) < 0$, $\forall v(t)$ such that $\lim\limits_{v \to \infty} \phi(v) = 0$. This is in accordance with the second modeling assertion.
3. ϕ admits the desired-velocity constraint $\phi(v^*) = s^*$ in accordance with the third modeling assertion.

5.6.3 Control Law and Stability Proof

The control input is the tangential acceleration $\dot{v}(t)$ updating the velocity v of which the state s is a function. We thus control combustion state via acceleration. The implication of $\phi(v^*) = s^*$ is that by tracking s^*, the torch is moved at the desired velocity v^*. Let $e_s(t) = s^* - s(v(t))$ be the combustion state error with respect to the desired state s^*. Its dynamics are: $\dot{e}_s(t) = -\frac{\mathrm{d}}{\mathrm{d}v}s(v)\dot{v}(t)$. Apply the control input $\dot{v}(t) = -ke_s(t)$ with $k > 0$ being a strictly positive gain. This results in the desired behavior of accelerating when s is excessive and decelerating when s is deficient. Substitute the control input in the dynamics equation to obtain: $\dot{e}_s(t) = ke_s(t)\frac{\mathrm{d}}{\mathrm{d}v}s(v)$. We prove stability using the Lyapunov candidate function $V(e_s) = \frac{1}{2}e_s^2$, computing

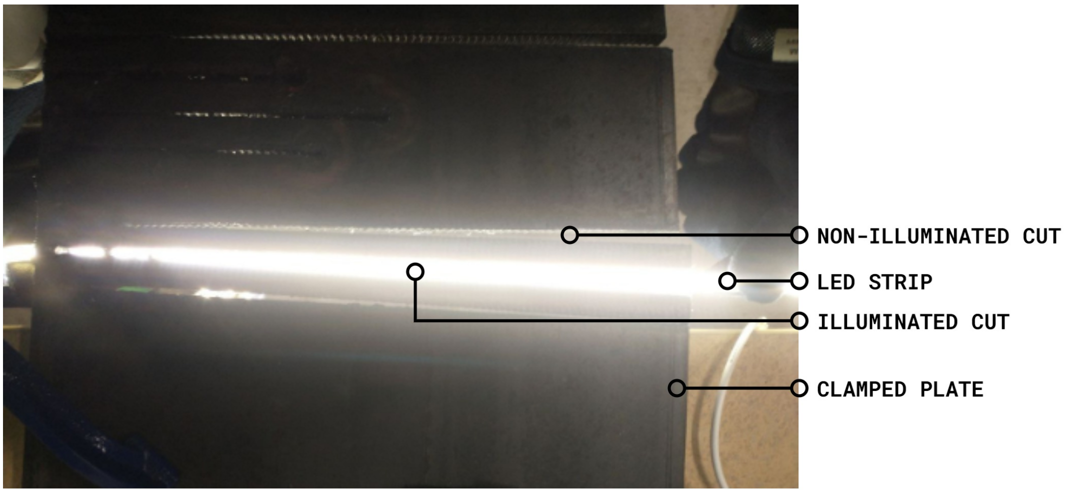


FIGURE 5.12: After a cutting experiment, a LED strip is placed along the cutting path to inspect the cut and its overall quality. © IEEE

its derivative:

$$\dot{V}\left(e_s(t)\right) = e_s\dot{e}_s = ke_s^2(t)\tfrac{\mathrm{d}}{\mathrm{d}v}s(v) < 0, \quad \forall e_s(t) \neq 0 \tag{5.3}$$

The previous result is true for all non-zero error states since $k > 0$, and $e_s^2(t) > 0$, and $\frac{\mathrm{d}}{\mathrm{d}v}s(v) = \frac{\mathrm{d}}{\mathrm{d}v}\phi(v) < 0$. The controlled system is thus asymptotically stable.

We note that the desired combustion state s^* is determined experimentally. Specifically, an adequate range of combustion states is found for a particular setup and would not change across cutting sessions. After a certain number of trials, a precise value for the desired state s^* is determined.

5.7 Autonomous Cutting Experiments

We evaluate our vision-based framework for autonomous cutting in physical cutting experiments performed using a 1-DOF robot, a cutting torch, oxy-propane gas, and steel plates. The physical setup is shown and annotated in Fig. 5.11. In our robotic setup, the camera is mounted at a fixed pose towards the torch tip. The camera lens is covered with a tinted visor used to dim the scene and focus on the flame, as is typically done by skilled cutters. This also prevents image saturation due to the extreme brightness of the flame and pool. The actuators are a stepper motor setting the torch's linear motion, and a linear actuator engaging the oxygen bypass. Using ROS [97], the system communications are distributed across a microcontroller for the low-level actuator control and a computer for image processing and high-level control. The computer is connected to the camera and the microcontroller, while the torch is connected to oxygen and propane tanks. The interaction of these components during the control loop is summarized in Fig. 5.10.

5.7.1 Experiment Workflow

The experiment workflow follows the oxy-fuel cutting sequence in Fig. 5.4. After setup, the experiment begins by igniting the cutting torch and setting the oxy-propane pressures; these are done manually. When an adequate torch flame is detected (as in

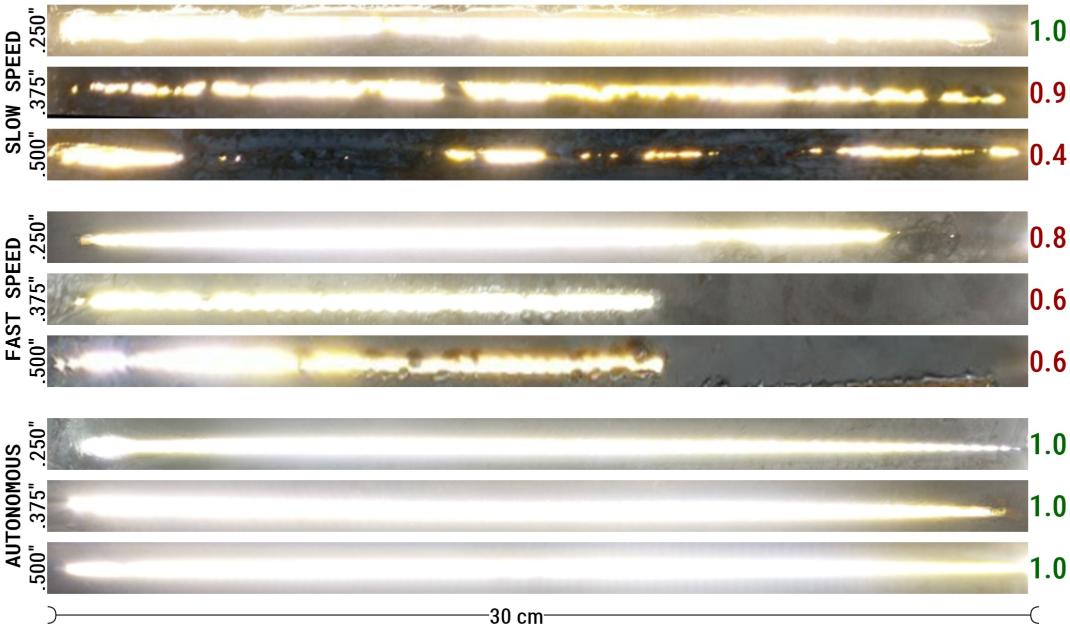


FIGURE 5.13: The illuminated cuts of each mode (right) for each plate thickness (right), the reference path length (bottom), and the cutting success ratio (left), defined as the length proportion of the path that is successfully cut, rounded to the nearest tenth. © IEEE

Section 5.5.1), calibration is started and the robot identifies the torch flame's centroid and its baseline intensity. Next, the robot moves the torch flame to the initial location on the metal plate to preheat the surface. Once the surface is sufficiently preheated (as in Section 5.5.2), the robot is entered into autonomous combustion control. The robot engages the oxygen bypass and autonomously sets the torch speed based on its vision-based control. In these experiments, the reference cutting paths for the robot are straight lines along the plate's surface. After cutting, the bypass is disengaged and the torch is turned off. A representative experiment is shown in Fig. 5.14.

5.7.2 Cutting Results

The experiments are performed on steel plates of thicknesses $(0.250, 0.375, 0.500)$ in. We conducted experiments with three cutting modes to compare the outcomes of our automated vision-based algorithm against that of constant torch speeds: (1) a constant torch speed of $0.2\,\mathrm{cm/s}$ (referred to as 'slow cuts'); (2) a constant torch speed of $3.2\,\mathrm{cm/s}$ (referred to as 'fast cuts'); and, (3) controlled speed determined by the algorithm. After the cut is performed on each plate by each mode, the plate is cooled down and each cut is illuminated for clear inspection by placing a LED strip behind the plate along the cutting path (see Fig. 5.12).

In addition, the controller's performance is examined in time-series plots (Fig. 5.15) wherein the following quantities are logged: (1) torch position, (2) torch velocity, (3) acceleration input, (4) pool convexity and intensity (and when combined, combustion state), and (5) their desired values. These are all plotted with normalized values. Pool convexity, intensity, and combustion state are normalized by default. Position is normalized by its maximum value reached at the cutting path's end. Velocity and acceleration are normalized by their maximum values v_{max} and $\dot{v}_{\mathrm{max}}$ imposed as safety constraints on the controller; though these values are never attained.

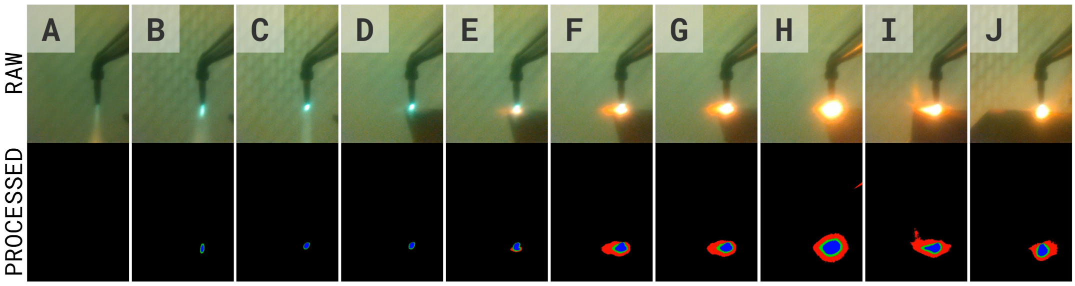


FIGURE 5.14: In this representative experiment, the robot autonomously calibrates the vision system, conditions the metal surface, and controls the torch motion to complete the cut on the steel plate. Key frames are labelled A through J to discuss important events throughout the experiment. © IEEE

All autonomous cuts are performed with these parameters: gain $k = 200$, pool convexity reference $c^* = 0.95$, pool intensity reference $i^* = 0.25$, maximum velocity $v_{\text{max}} = 2.0\,\text{cm/s}$, maximum acceleration $\dot{v}_{\text{max}} = 0.8\,\text{cm/s}^2$. While the problem is noisy, the controller converges in every case and successfully tracks the desired combustion state.

5.7.3 Discussion of Autonomous Experiments

Referring to Fig. 5.14, the robot autonomously performs the cut as follows. As the torch pressures are being configured (frames A through C) the visual feedback is monitored and the calibration predicates are tracked. Once the flame stabilizes (frame C) and all calibration predicates are true, the vision system is calibrated, and the robot moves the torch flame to the initial cut location (frame D) for preheating. The heat pool forms (frame E) and gradually grows (frame F) until all conditioning predicates are true (frame G). The robot then begins the combustion phase (frame H) and controls the torch motion (frames I and J) until the cut endpoint is reached.

5.7.4 Discussion of Cutting Results

Referring to Fig. 5.13, one may intuitively expect the slow cuts to fully penetrate each plate along the cutting path. In actuality, moving the torch too slowly accumulates heat excessively in regions surrounding the cutting path thereby re-sealing portions of the cut with creeping deposits of heated steel (cut success ratios 0.4 and 0.9). Note that the 0.250 in. plate is thin enough that the excess heat combusts these surrounding regions yielding the wide cut (ratio 1.0). Nevertheless, slow cuts are wasteful in time and resources in addition to yielding substandard cut textures. The fast cuts instead tend to extinguish the heat pool before the cut's completion (ratios 0.6 and 0.8) due to diminishing heat accumulation, fault of moving the torch too quickly. The effects of moving too slow or too fast are aggravated on thicker plates due to a larger material volume surrounding the cutting path. In our vision-based formulation, these faults translate to combustion states that are excessive (slow cuts) or deficient (fast cuts).

By contrast, the autonomous cuts are clear and successful in every case (ratios 1.0). Without knowing the plate thicknesses, the controller adjusts the torch velocity towards a stable region to converge and maintain the combustion state near its desired value (see Fig. 5.15). In effect, Fig. 5.13 shows the resulting successful cuts against each of the plate thicknesses.

5.8 Conclusion

In this chapter, we formalize the oxy-fuel cutting problem and develop a vision-based framework for autonomous cutting. The vision feedback describes the heat pool's combustion state using its convexity and intensity. We devise predicates for autonomously calibrating the vision system and conditioning the metal surface. Afterwards, our vision-based controller updates the torch velocity along the cutting path. Our experiments demonstrate the successful autonomous cutting of steel plates (varying thicknesses) with a 1-DOF robot using purely visual feedback.

While our framework addresses autonomous combustion cutting, further work is required to evaluate our methods on 2D and 3D cuts. Furthermore, this approach is limited to combustion cutting (where the metal is predominantly burned, *e.g.*, with carbon steel) and does not directly address the less common fusion cutting (where the metal is predominantly melted, *e.g.*, with aluminum).

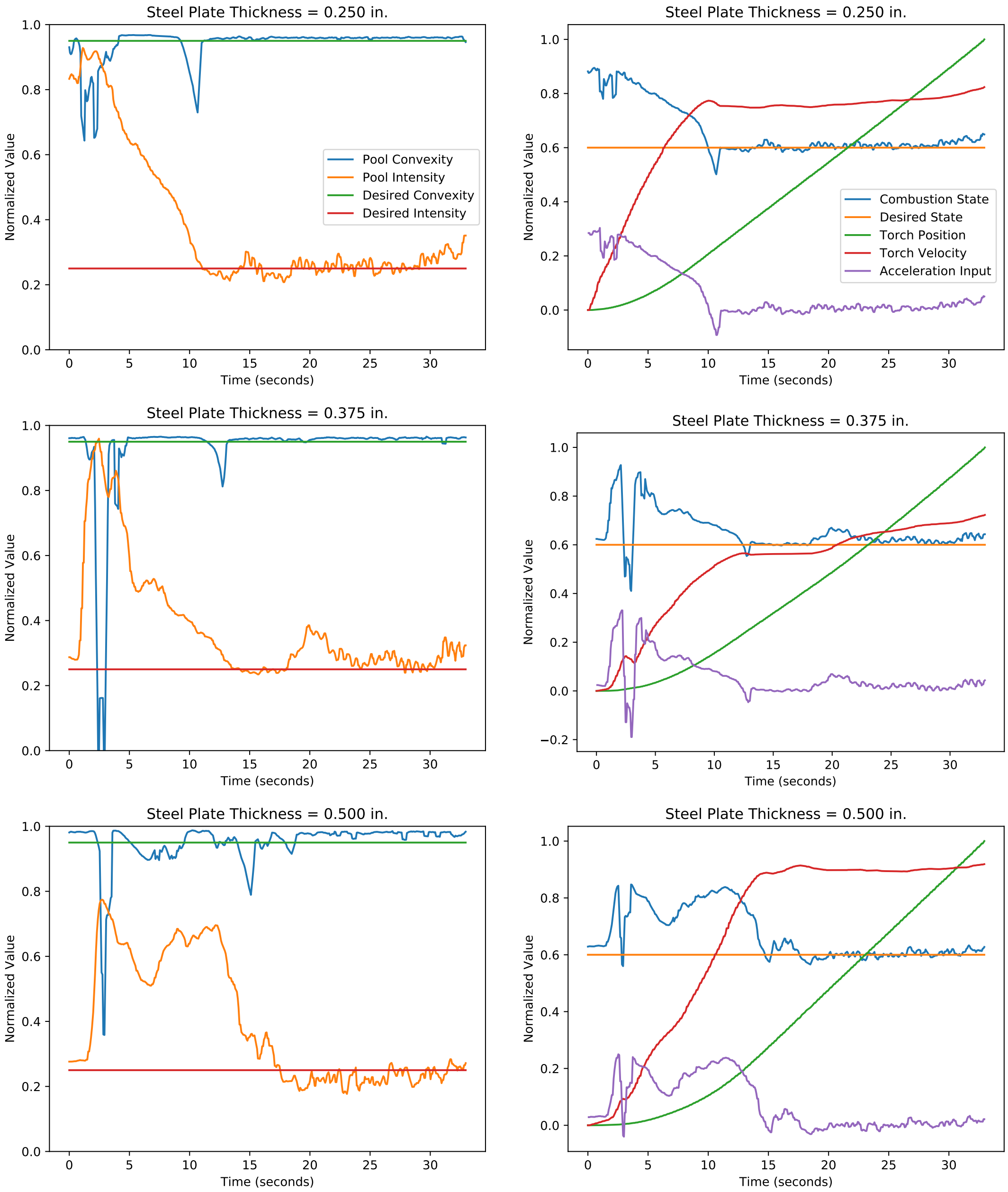


FIGURE 5.15: Time-series plots of the autonomous experiments (combustion cutting phase) for each plate. The left column row shows both pool features, whereas the right column combines them into combustion state. The legends for each column are on the topmost row. © IEEE

Chapter 6

Task State Monitoring

THE vision-based framework developed in Chapter 5 restricts the robot actions to task states tailored to the oxy-fuel cutting operation. This chapter focuses on monitoring these task states for the purpose of either validating the robot actions, or detecting anomalies. The industrial operation of oxy-fuel metal cutting via gas torches involves tasks such as ignition, preheating, and combustion along the target surface. Automated oxy-fuel cutting systems are exposed to risks and anomalies that can lead to incorrect actions and safety hazards. In this chapter, we develop a classifier for online task state estimation to assess the cutting robot's actions, detect anomalies, and reduce the risk of hazards. Using representative footage from our robotic cutting experiments, we curate an image dataset labeled with four types of cutting task states. Using deep learning methods, we design and train a convolutional neural network model for classifying the cutting task state from input images. The classifier architecture is optimized for rapid inferences during online estimation. After evaluation, our classifier achieves an overall accuracy of 93.8% with high inference speeds on two types of representative hardware. Our 'Oxy-fuel Cutting Task State' (OCTS) dataset is available at doi.org/10.5281/zenodo.7734951.

Keywords: *Task state estimation, anomaly detection, oxy-fuel cutting, deep learning, convolutional neural networks, image dataset, multinomial classification*.

Notice: The work in this chapter is accepted at the 2023 IEEE International Conference on Automation Science and Engineering (CASE) and is subject to IEEE Copyright.

6.1 Introduction

The global economy of the future is projected to become increasingly automated. The industrial sector is particularly subject to pro-automation market pressures such as falling prices of automation systems [141]. However, scaling up the automated economy comes with its unique challenges. In particular, the rising importance of workplace safety [142] imposes stricter safety requirements in the design and adoption of automated systems [143]. This is emphasized in the automation of hazardous work that is highly-exposed to risk, such as metal cutting and welding. In parallel, there is considerable incentive to automate the difficult processes of oxy-fuel cutting [118] and welding. In existing work, the focus is often to achieve some autonomous functionality while safety is not sufficiently addressed.

In this chapter, we develop a classifier for task state estimation to monitor automated oxy-fuel cutting systems and detect anomalies for the benefit of increasing workplace

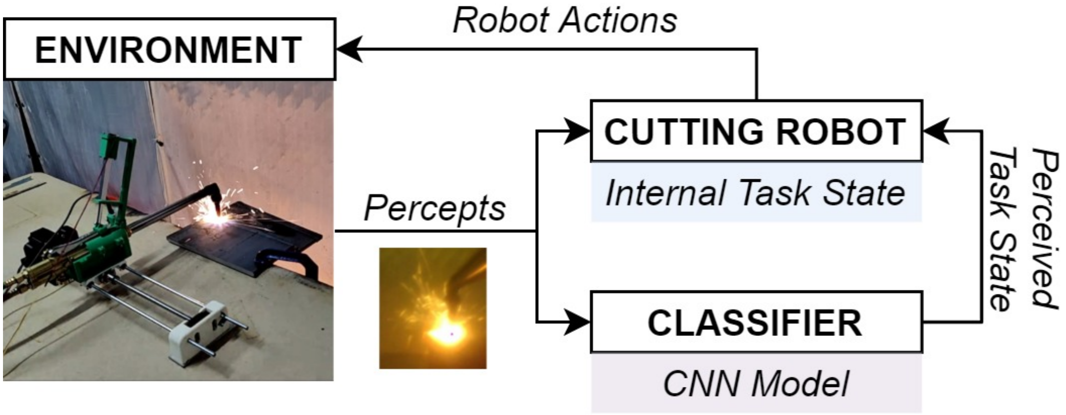


FIGURE 6.1: The classifier interprets the environment percepts and provides the perceived task state to the cutting robot. When the internal and perceived task states match, the robot actions are validated and executed. Upon disagreement, an anomaly is declared and cutting operations are halted. © IEEE

safety. For this, we curate an image dataset of oxy-fuel metal cutting scenarios, representative of key task states in oxy-fuel cutting. The images are obtained from a series of recorded cutting experiments performed with a torch-equipped robot that cuts steel plates. These images are then labelled into distinct task state classes, identified by their dominant visual feature ('Torch flame', 'Preheating pool', 'Combustion pool', and 'Not applicable'). Using this data, we adopt a deep learning approach for capturing the data's feature hierarchy via a convolution neural network (CNN) model.

The purpose of this classifier is to monitor live oxy-fuel cutting operations. The classifier receives online vision data and infers the cutting task state, providing an interpretation to the robot (Fig. 6.1). When the classifier's interpretation and the robot's internal task state agree, an anomaly is unlikely and the robot proceeds with its actions. Conversely, a discrepancy between them suggests the occurrence of an anomaly in which case cutting operations are halted for safety inspection and response. The aim is to help validate the robot's actions for the detected task state. By monitoring the task environment and signaling discrepancies with the robot's actions, the classifier adds one layer of safety.

The core contributions of this chapter are:

1. Creating an image dataset for oxy-fuel cutting task states, obtained from automated cutting experiments.

2. Developing a CNN model to classify the task state from input images with high inference speed.

3. Evaluating the classifier's performance against a separate test set previously unseen by the classifier and identifying its strengths and limitations.

To the best of our knowledge, this is the first study in the literature focusing on vision-based state estimation of oxy-fuel cutting operations. Furthermore, the work in this chapter publishes the first dataset [144] in the literature covering the task states of oxy-fuel cutting and the first treatment in developing an appropriate classifier for these task states.

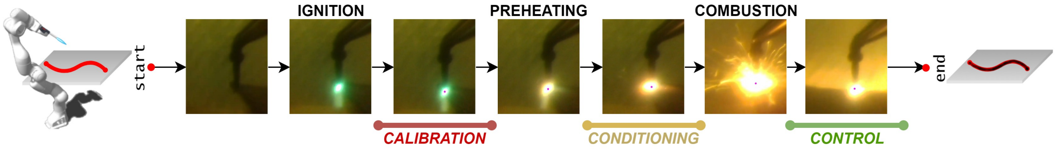


FIGURE 6.2: Sequence of events (labeled in black) and tasks (labeled in color) in vision-based autonomous oxy-fuel cutting. Conceptually, the robot is equipped with a vision sensor and cutting torch and must cut the target object along the desired cutting path. The cutting events are the instances that trigger the cutting tasks. The footage shown is retrieved from vision-based cutting experiments using our 1-DOF cutting robot. © IEEE

6.2 Related Work

Industrial processes using welding, cutting, and laser tooling are studied using a variety of instrumentation and techniques to extract higher-level perceptual information from lower-level sensory data. Often, the processed area (which may exhibit a heat pool or melt pool) is monitored and characterized to model or predict its effect on process quality, anomaly and defect detection, or penetration depth. Non-learning based techniques can be used to characterize combustion such as using spectral analysis [145] or for the efficient operation of industrial furnaces using classical image processing [146] and 3D instrumentation [147]. However, these approaches are often limited to processes that can be explicitly modeled.

More recent advances focus on applying learning-based approaches to infer task-relevant information while generalizing to a broader range of input scenarios. Indeed, the aim of learning-based applications to is to enable or improve industrial processes by monitoring for errors and ensuring that the goal states are reached. Such data-driven and learning-based methods are successfully applied to a wide variety of media: gas tungsten arc welding (GTAW) [148]–[152], gas metal arc welding (GMAW) [153], [154], submerged arc welding [155], variable polarity plasma arc welding [156], laser welding [157]–[159], wire-arc additive manufacturing (WAAM) [160], [161], wirefeed laser additive manufacturing [162], and orthogonal metal cutting [163].

In effect, neural network models are used abundantly for such industrial processes. While diverse neural architectures are adopted, convolutional neural networks [150], [151], [154], [157], [159], [161], [162], [164], [165] are most encountered in these applications. Often, pre-trained models are used or adapted such as the ResNet architecture for its deep convolutional layers and residual connections [158], [160]. Other approaches applied include generative adversarial networks [153], ensemble methods [152], autoencoders [163], vision transformers [149], transfer learning [148], and extreme learning machines [156].

In practice, while the problems encountered can exhibit similarities, they also carry distinct challenges due to the particularities of their tooling and the differences in their data. These domain-specific distinctions not only suggest preference for certain learning models, but also inform about auxiliary techniques that exploit domain knowledge. For instance, X-ray imaging can reveal more information for certain weld defects [159], [164]. Similarly, problem-specific advantages arise with other auxiliary techniques such as multimodal sensing [153], [157], [160], multisource sensing [152], image preprocessing [148]–[151], and acoustic sensing [154].

The following problems in prior work are most related to ours. For GTAW and weld pool image data: penetration classification (3 classes) in [152] and prediction in [151]; pool classification (2 classes, 6 classes) for defect identification in [148]; and, penetration state classification (4 classes) for penetration recognition in [149]. For GMAW and weld pool image data: pool state classification (4 classes) for defect detection in [153]. For WAAM and melt pool image data: pool state classification (4 classes) for anomaly detection in [161].

While there are similarities, our problem is distinct for the following reasons:

1. The industrial process is oxy-fuel metal cutting producing a heat pool from combustion, not a weld pool or melt pool.
2. The image data is particular to the oxy-fuel cutting medium and its associated events and tasks.
3. The problem is to classify the cutting task state for monitoring the robot actions' safety and correctness.
4. The classifier's inference time must be sufficiently low for online estimation, thus constraining its design and architecture.

Moreover, the aim of task state estimation is to enable higher-level reasoning about the robot's actions from lower-level data. For instance, [166] applies this strategy to robot contact tasks, extracting high-level action grammars from low-level trajectory data. To the best of our knowledge, this is the first work applying deep learning to automated oxy-fuel cutting for classifying its task states.

6.3 Problem Formulation

This section delineates the problem elements: the process and tooling of oxy-fuel cutting, the events and tasks relevant to its automation, and the role of task state estimation.

6.3.1 Oxy-fuel Cutting

The operation of oxy-fuel cutting consists of manipulating a cutting torch along a metal surface to cut through it along a desired path. The torch flame is produced by burning an oxy-fuel gas mixture, where the fuel is typically acetylene or propane. Material removal is usually achieved via a combustion reaction with the metal (most commonly carbon steel). This requires the metal surface to be sufficiently preheated using the torch flame. During preheating, the heated region on the metal surface exhibits the formation of a heat pool. This apparent bright blob is an accumulation of heat where combustion is most intense. Upon sufficient preheating, combustion is intensified by increasing oxygen flow and the torch is moved along the desired cutting path at an adequate velocity to maintain adequate conditions for combustion and material removal.

6.3.2 Automated Cutting

The oxy-fuel cutting process is complex and involves several events and tasks. Its automation may be tackled via different approaches, depending on the sensing modality and the desired degree of autonomy. We focus on vision-based automated cutting due to the visual stimuli produced by the heat pool and the vision-based tracking inspired by the techniques of skilled cutting workers. This requires a representation of the cutting problem tailored to the vision-based approach of its automation.

The sequence of events and tasks during vision-based automated oxy-fuel cutting is illustrated in Fig. 6.2. The key events during the cutting process are: (1) *Ignition*: The torch flame is ignited and focused; (2) *Preheating*: The flame is positioned at the surface; and, (3) *Combustion*: Oxygen flow is increased via a lever. Accordingly, the cutting system executes these tasks: (1) *Calibration*: Calibrate the vision system against the torch's flame; (2) *Conditioning*: Heat the surface to combustion conditions; and, (3) *Control*: Regulate the torch motion for combustion cutting.

6.3.3 Task State Estimation

The above sequence of tasks for vision-based cutting, while effective, executes under the assumption of ideal operational conditions. When the expected cutting event is detected, the task state is updated and the robot continues onto its next action. In practice, oxy-fuel cutting operations are exposed to various potentials errors, risks, and hazards.

We enumerate instances of potential failure modes: ignition may fail; the flame may not be correctly focused; calibration may fail due to excess noise; combustion may fail fault of insufficient conditioning; faulty oxy-fuel tooling and leaks; excess sparks, slag, fire, or other anomalies; electronic or mechanical failures, and software errors; among others.

Such failures would manifest as anomalies in the image frame and the heat pool. While the automated system possesses internal state of its actions, it lacks external state of its environment. As such, under anomalous conditions the cutting agent's actions may be incompatible with the state of its environment. Under such cases, there may occur inefficiencies, failures, hazards, and dangers without intervention.

To address this, the cutting system tasks must be monitored and validated. This can be achieved using a classifier that infers from the robot's visual input one of four task states identified by their dominant feature in the image: 'Torch flame', 'Preheating pool', 'Combustion pool', and 'Not applicable'. This classifier is trained on representative footage, labeled with the desired task states. During cutting, the robot would check its internal task state with the classifier before taking action (see Fig. 6.1). Using this task state estimation, the robot gains some external awareness and performs safer cutting operations.

6.4 Oxy-fuel Cutting Dataset

To train and test the task state classifier, we develop a dedicated dataset that captures the particularities of automated oxy-fuel cutting. This need is reinforced due to the lack of relevant datasets that are publicly available. Our 'Oxy-fuel Cutting Task State' (OCTS) dataset can be accessed at `doi.org/10.5281/zenodo.7734951`.

6.4.1 Data Collection from Cutting Experiments

Using our automated setup shown in Fig. 6.3, we perform cutting operations (as in Fig. 6.2) on steel plates. The area of interest on plates is recorded using the robot's eye-in-hand RGB camera. In total, 50 experiments across 11 sessions are recorded using the Intel RealSense D435 yielding 142,671 collected images. Some cuts are

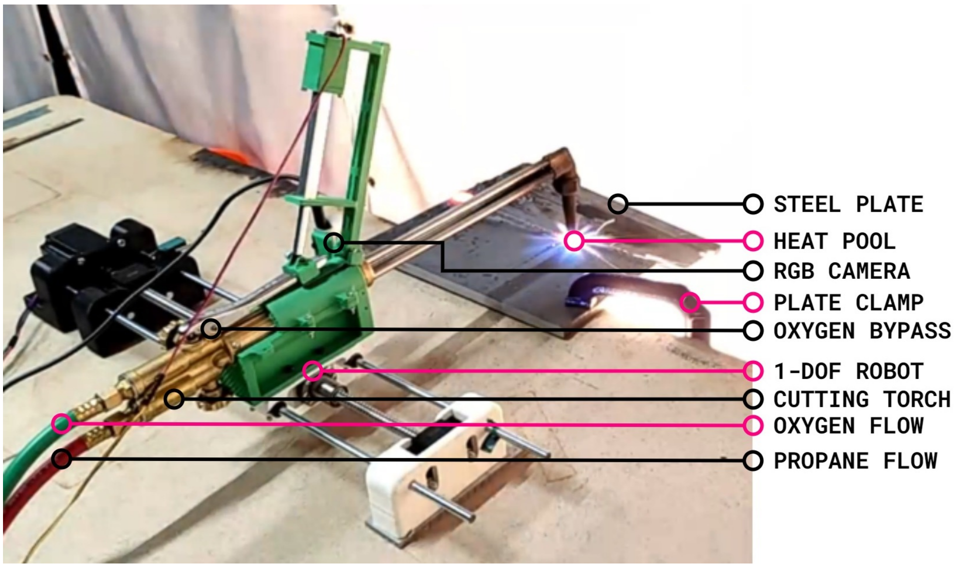


FIGURE 6.3: The 1-DOF cutting robot performs cutting experiments wherein its RGB camera records the affected area on the steel plates. We note that the camera is mounted at a fixed pose towards the torch tip and that its lens is covered with a tinted visor (as is worn by skilled cutters). This tinted visor dims the scene, focuses on the flame and the pool, and prevents image saturation due to the extreme brightness of the flame and the pool. © IEEE

TABLE 6.1: Summary of the Data Collection Parameters © IEEE

Total Experiments	50	**Frames Collected**	142671
Footage Recorded (min)	119	**Frame Dimensions** (px)	(640, 480)
Recording Rate (fps)	20	**Frame Channels**	(R, G, B)

completed without fault (Fig. 6.4), while others are subjected to diverse failure modes. Depending on the severity of the failure, cuts are either fully completed, or partially completed due to interruptions. In this manner, the footage spans a broad range of representative, varied, and diverse cutting conditions. The data collection parameters are summarized in Table 6.1.

6.4.2 Data Labels & Classification Problem

The classification problem is based on identifying the prominent element in the image: the torch flame (TF), the preheating pool (PP), or the combustion pool (CP). In addition, the label NA corresponds to 'Not applicable' where no dominant element is identified. This indicates cutting conditions outside the range of normalcy and is treated as an anomalous state. The cutting task state in an image is identified by its associated dominant element as summarized in Table 6.2. When the dominant element in the image frame is the torch flame (TF), the nominal task is to calibrate the vision system; when it is the preheat pool (PP), to condition the surface; and when it is the combustion pool (CP), to control the torch motion. When no dominant element is found (NA), the corresponding task is to halt cutting operations for inspection and response. In addition, a mismatch between the detected element (perceived task state) and the robot's intended action (internal task state) constitutes anomalous behavior.

TABLE 6.2: Summary of Labels and their Associated Task States © IEEE

Label	Element	Nominal Task State
TF	Torch flame	Calibrate the vision system.
PP	Preheating pool	Condition the metal surface.
CP	Combustion pool	Control the torch motion.
NA	Not applicable	Halt the cutting operations.

Sample data for each class is also shown in Fig. 6.5. As such, the problem is structured as a multinomial classification with four labels: $\{\texttt{TF}, \texttt{PP}, \texttt{CP}, \texttt{NA}\}$. In our dataset, each image is labeled with one of these four labels according to the prominent element in the image and its associated task state.

6.5 Model Design

In this section, we detail the architecture of our CNN-based model and its input data's preprocessing.

6.5.1 Model Architecture

The purpose of the model is to classify input images into one of the aforementioned four cutting task states. The subject and contents of the image data incorporate a large variety of geometrical and chromatic interrelationships between the features of the torch flame, the heat pool, and the surrounding environment. Accordingly, it is non-trivial to express its intricate feature hierarchy via explicit feature engineering. When considering the various model choices for image classification [167] and their respective advantages, we find that CNN models are adequate for our problem given their established capabilities for feature engineering.

Additionally, given our constraints on inference speed for online monitoring requirements, we consider the effects of design parameters on neural network expressivity and performance [168], [169]. In particular, we desire a CNN architecture that is large enough for enabling the model to express a sufficient amount of variations within the data and thus yield a satisfactory prediction accuracy. At the same time, the model's architecture must be small enough for rapid inference during online monitoring.

By factoring this tradeoff between prediction accuracy and inference speed during design iterations, we adopt the architecture shown in Fig. 6.6. Our neural network

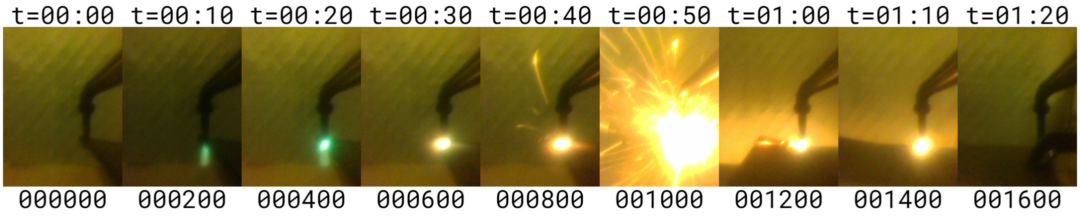


FIGURE 6.4: Sequential footage from a particular experiment is shown at intervals of 200 frames (or 10 second). This particular cut is executed without fault. © IEEE

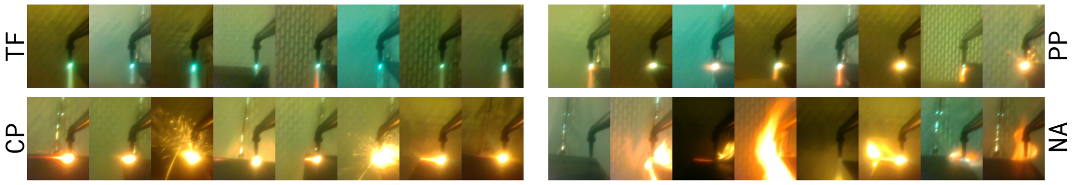


FIGURE 6.5: Sample frames for each class from various experiments. We note the importance of covering variations of the key elements and ambient conditions. © IEEE

model $\hat{\mathbf{y}} = f(\mathbf{X}, \boldsymbol{\theta})$ maps the input $\mathbf{X} \in \mathbb{R}^{3\times 640\times 480}$ which are 3-channel RGB images of size 640×480, to the output $\hat{\mathbf{y}} \in \mathbb{R}^4$ which are the class scores, given the mapping parameters $\boldsymbol{\theta}$, *i.e.*, the model weights. Our network f (expressed in functional form) is composed of four functional blocks,

$$f = \mathsf{B}_4 \circ \mathsf{B}_3 \circ \mathsf{B}_2 \circ \mathsf{B}_1. \tag{6.1}$$

At the input, we begin with two convolutional functional blocks B_1 and B_2 composed as follows:

$$\begin{aligned} \mathsf{B}_1 &= \texttt{Pool}_{11} \circ \texttt{Drop}_{11} \circ \sigma \circ \texttt{Conv}_{12} \circ \sigma \circ \texttt{Conv}_{11} \\ \mathsf{B}_2 &= \texttt{Pool}_{21} \circ \texttt{Drop}_{21} \circ \sigma \circ \texttt{Conv}_{22} \circ \sigma \circ \texttt{Conv}_{21} \end{aligned} \tag{6.2}$$

Here, `Conv` are convolutional layers, `Pool` are max-pooling layers, `Drop` are dropout layers (for regularization), σ is the Sigmoid Linear Unit (SiLU) activation function, and the operator $\circ$ is function composition. The blocks B_1 and B_2 are followed by a fully-connected functional block B_3,

$$\begin{aligned} \mathsf{B}_3 = \; &\texttt{Drop}_{33} \circ \sigma \circ \texttt{Dense}_{33} \circ \texttt{Drop}_{32} \circ \sigma \circ \texttt{Dense}_{32} \circ \\ &\texttt{Drop}_{31} \circ \sigma \circ \texttt{Dense}_{31} \circ \texttt{Vec} \end{aligned} \tag{6.3}$$

where `Vec` vectorizes its input into a 1D vector, and `Dense` are fully-connected layers. Finally, the output block $\mathsf{B}_4 = \texttt{Dense}_{41}$ contains a single dense layer. The hyperparameters of each block's layers as well as the transformation of the data along the architecture are indicated in Fig. 6.6.

6.5.2 Data Preprocessing

Within our image data, the regions that are more pertinent and crucial are those containing information about the torch flame or heat pool. Accordingly, regions

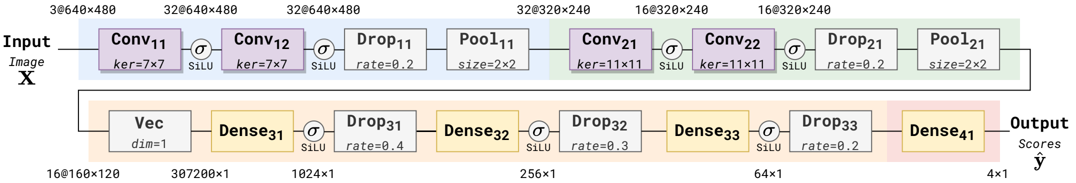


FIGURE 6.6: The architecture of our neural network. Our model is composed of two convolutional blocks (blue and green), one dense block (orange), and one output block (red). The respective hyperparameters of each block are indicated therein and the state of the data (channels, rows, and columns) are displayed along its mappings between layers. The stride of all convolutional layers is $(1, 1)$. All activations are Sigmoid Linear Unit functions. © IEEE

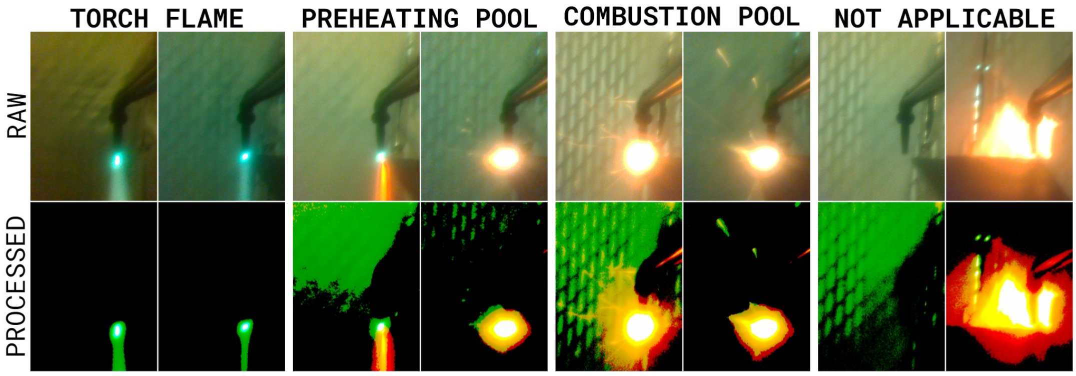


FIGURE 6.7: The preprocessing is demonstrated on samples from each class. © IEEE

within the image that are relatively brighter are of higher interest, whereas those that are relatively dimmer contain information that is less relevant to our task, *i.e.*, noise. For these reasons, we process the input images (Fig. 6.7) using thresholding to nullify regions of low interest. At the same time, the thresholding must preserve the desirable bright regions in the image. For instance, binary-thresholding (resulting in a binary image) would lose much of the essential information in the bright regions required to reliably discriminate between scenarios. As such, we adopt a one-sided channel-wise thresholding method as specified in the procedure `ProcessImage` below.

```
define ProcessImage(image $\mathbf{X}$):
    $(\alpha_R, \alpha_G, \alpha_B) \leftarrow (175, 150, 225)$
    for each channel $c \in \{R, G, B\}$ do
        for each pixel $(i, j) \in \mathbf{X}_c$ do
            if $\mathbf{X}_c[i,j] < \alpha_c$ then $\mathbf{X}_c[i,j] \leftarrow 0$
    return $\mathbf{X}$
```

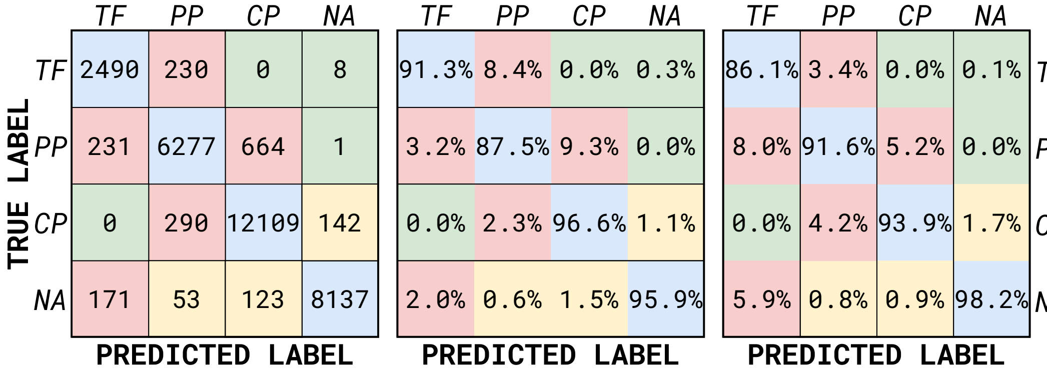
In each individual channel $c \in \{R, G, B\}$, the pixel values are zeroed when they fall below the thresholds α_c, and unchanged otherwise. By performing exploratory data analysis on the training data, we determine the threshold values $(175, 150, 225)$ to sufficiently eliminate noise while preserving relevant information. We note that thresholds are applied channel-wise since the proportion of noise is different across channels, motivating distinct values for α_c.

6.6 Training & Evaluation

We delineate our model's training and evaluation below.

6.6.1 Model Training and Data Augmentation

Our dataset comprises of 50 experiments recorded at different times-of-day. We split our data into training and testing sets experiment-wise, *i.e.*, we consider 38 recorded experiments for training and 12 for testing, yielding a train-test split of around $(78\%, 22\%)$ or $(111745, 30926)$ images. Through this mutually-exclusive sampling from experiments, we emphasize the learning of key features in the torch flame and heat pool across all experiments and discourage the model from learning the noise, conditions, and background effects unique to each experiment. Additionally for training, we sample our experiment sets across the range of times-of-day to train the model



TRUE LABEL	TF	PP	CP	NA
TF	2490	230	0	8
PP	231	6277	664	1
CP	0	290	12109	142
NA	171	53	123	8137

PREDICTED LABEL

	TF	PP	CP	NA
TF	91.3%	8.4%	0.0%	0.3%
PP	3.2%	87.5%	9.3%	0.0%
CP	0.0%	2.3%	96.6%	1.1%
NA	2.0%	0.6%	1.5%	95.9%

PREDICTED LABEL

	TF	PP	CP	NA
TF	86.1%	3.4%	0.0%	0.1%
PP	8.0%	91.6%	5.2%	0.0%
CP	0.0%	4.2%	93.9%	1.7%
NA	5.9%	0.8%	0.9%	98.2%

PREDICTED LABEL

FIGURE 6.8: Confusion matrices of the model's prediction on the test dataset. The diagonals indicate the correct predictions whereas the off-diagonal terms indicate misclassifications. Red, yellow, and green cells indicate label-wise concern that is major, minor, and negligible respectively. **Left:** The matrix is expressed with absolute count. **Center:** The matrix is expressed relative to the true label totals (row-by-row), thereby showing class recall on the diagonal. **Right:** The matrix is expressed relative to the predicted label totals (column-by-column), thereby showing class precision on the diagonal. © IEEE

on varied ambient conditions (*e.g.*, lighting, temperature) that are affected by the time-of-day. To further increase our model's robustness against noise in our dataset, we apply several randomized techniques for augmenting our training data. Specifically, we introduce subtle variations in the images by randomizing the following image parameters: shearing, brightness, contrast, saturation, and Gaussian blurring. During training, we minimize the empirical risk as in $\min_\theta \frac{1}{N}\sum_k^N L[\mathbf{y}_k, f(\mathbf{X}_k, \theta)]$ where L is the categorical cross-entropy loss function, N is the number of training samples, $\mathbf{X}_k$ is the k^{th} training input, and $\mathbf{y}_k$ is its true label. While there are many suitable optimization methods [170] for this minimization problem, we use the adaptive moment estimation (Adam) algorithm [171].

6.6.2 Model Testing and Evaluation

After testing on the 12-experiment test set, we evaluate our model to assess its output's validity and coherence with our modeling assumptions. Our evaluation metrics for this state classification task are overall model accuracy (Table 6.3) both on the entire test set and its individual experiments, as well as the confusion matrices (Fig. 6.8) showing class-wise prediction count, recall, and precision. We thus assess our assumption that the task states {TF, PP, CP, NA} are visually distinguishable in our dataset, which would be reflected as sufficiently large diagonal terms in the confusion matrix.

6.6.3 Results and Discussion

In general, our model performs reliably, achieving an overall accuracy of 93.814% (Table 6.3). Moreover, its accuracy on individual test sets ranges from 88.725% and above, and lies within [90.574%, 96.665%] for two-thirds (8/12) of them. In particular, it distinguishes the NA class quite well (with a precision of 98.2% and a recall of 95.9%) despite its intra-class variation (see Fig. 6.5) suggesting that it can confidently identify anomalous conditions during cutting.

We study the misclassfications (Fig. 6.8) of concern deemed major (red, 2% or more), minor (yellow, within [0.5%, 2%)), and negligible (green, less than 0.5%) expressed relative to the true label totals (center matrix) and predicted label totals (right matrix). Major concern occurs mainly during inter-class transitions, for instance, when the cutting task transitions from having a torch flame (TF) to preheating (PP), or from preheating (PP) to combustion (CP). In addition, the NA class is concentrated before ignition (no torch flame) and after combustion (the heat pool and torch flame gradually extinguish) wherein no dominant element can be identified; these constitute transitions from NA to TF and from CP to NA. Confusion during transitions is reasonable since even human experts struggle to distinguish these transitional images. More so, our results suggest that each transition should have its own respective class and thus more refined labeling.

For assessing inference speed, we evaluated our model on the test set using two representative types of hardware: a special-purpose GPU (NVIDIA A100 Tensor Core) and a mid-range consumer CPU (Intel Core i5-5257U). These respectively represent optimistic and pessimistic estimates of the inference speed, depending on the deployed model's computing hardware. We compute the average time of inferring one image at a time across all test images, resulting in an average inference time of 1.46 ms (GPU) and 1.25 s (CPU). In either case, the model's inference is sufficiently fast for prompt anomaly detection and safety response.

6.7 Conclusion

In this chapter, we develop a task state classifier for improving the safety of automated oxy-fuel cutting. For this, we curate a labeled dataset by conducting automated cutting experiments using a 1-DOF robot. Our CNN-based model is composed of four functional blocks (each containing its own layers): two convolutional blocks, one dense block, and one dense output block. We preprocess the inputs using one-sided channel-wise thresholding to eliminate noise and preserve the desirable image contents. We train and evaluate our model and achieve an overall accuracy of 93.814% with sufficient average inference speed on both a high-end GPU (1.46 ms) and a mid-range CPU (1.25 s). Nevertheless, our model struggles with inter-class transitions motivating the need for more refined classes in future work.

TABLE 6.3: Evaluation results showing the model's overall accuracy and class-wise recall against the testing data © IEEE

Test Set	Total Inputs	Correct Outputs	Model Accuracy	TF Recall	PP Recall	CP Recall	NA Recall
S13	312	312	100.00	—	—	—	100.00
S19	4806	4569	95.069	53.846	100.00	99.242	96.613
S21	2482	2365	95.286	97.541	95.602	94.953	95.210
S25	2909	2581	88.725	100.00	68.916	87.520	97.978
S29	1509	1498	99.271	100.00	—	—	99.265
S34	2472	2239	90.574	97.020	77.215	98.535	94.754
S36	2984	2716	91.019	99.539	78.980	97.853	98.361
S38	2321	2060	88.755	95.556	73.849	97.306	90.022
S40	5277	5101	96.665	97.802	99.208	96.097	92.745
S45	2682	2586	96.421	99.689	96.712	98.419	91.605
S47	1532	1426	93.081	97.458	79.076	98.000	97.073
S49	1640	1560	95.122	98.537	90.864	98.792	95.943
OVERALL	30926	29013	93.814	91.276	87.509	96.555	95.910

Note: Class recalls marked with "—" indicate no data instances of the corresponding class in the particular experiment set.

Chapter 7

Safe Structural Disassembly

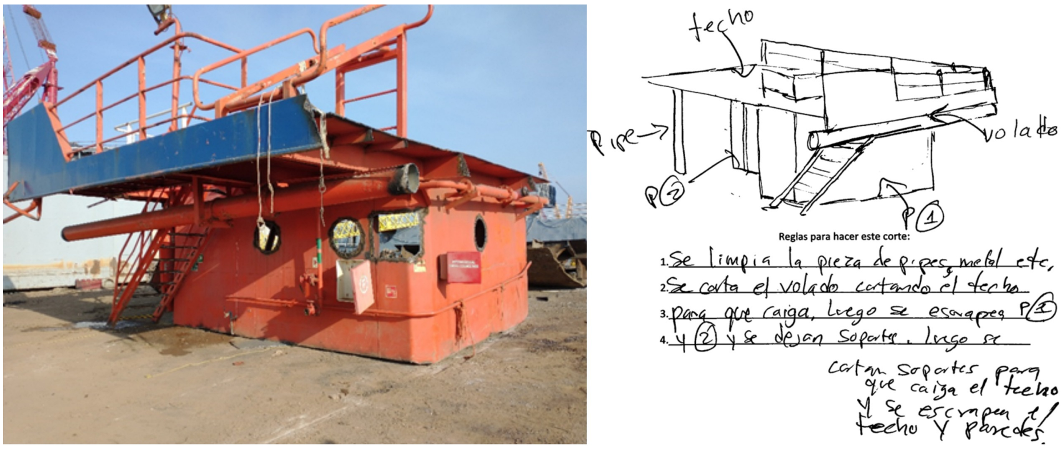
WHILE in Chapters 4–6 the focus is the cutting operation itself—its path, execution, and monitoring, respectively—instead, this chapter focuses on the higher-level problem of sequencing cuts on large structures for safely disassembling them. Disassembly and fragmentation are key operations in the dismantling and recycling of decommissioned structures such as aircraft, vessels, and buildings. Often, such operations are hazardous requiring careful planning for safe execution based on the experience and intuition of workers and forepersons. We propose and devise an algorithm for the automated sequencing of cuts to disassemble large structures. Using feedback from physics-based simulations and a mathematical model for safety, our algorithm performs sequential decision-making yielding the order of the cuts on the structure and the corresponding safe standing positions of the cutter (representing a worker or a robot). Our goal is to determine a sequence of cuts and cutter locations to maximize safety for the cutter and the environment. We establish the optimal solution via exhaustive searching, and design a greedy decision scheme to reduce the search runtime. Using our evaluations in simulation, we compare our greedy decision scheme against exhaustive searching and random searching, concluding that it satisfices the goal with high safety scores and low runtime.

Keywords: *Structural disassembly, sequential decision-making, algorithmic cut sequencing, safety modeling, physics-based simulation.*

Notice: The work in this chapter is accepted at the 2023 IEEE Conference on Decision and Control (CDC) and is subject to IEEE Copyright.

7.1 Introduction

In contemporary industry, priority is often given to activities of synthesis such as production, assembly, construction, and packaging. These can be contrasted with activities of fragmentation such as dismantling, deconstruction, demolition, and unpacking. In light of the Circular Economy [172] and sustainable development [173], there is a growing importance of the end-of-life handling of products and structures, such as refurbishing and material recovery. In effect, the processes of disassembly, deconstruction, dismantling, and recycling are of particular interest to decommissioned vessels [21], aircraft [174], [175], buildings [176], offshore platforms [177] or other large structures. However, the disassembly of such large structures involves difficult and hazardous operations often citing concerns over occupational safety and environmental impact as in the ship recycling industry [178], [179]. Much of these hazards and liabilities arise due to the economic incentives on both the demand and supply sides. As discussed in [180], companies wishing to dispose their aging



Reglas para hacer este corte:

1. Se limpia la pieza de pipes, metal etc,
2. Se corta el volado cortando el techo
3. para que caiga, luego se escrapea P(1)
4. y (2) y se dejan Soportes. luego se
cortan soportes para que caiga el techo y se escrapean el techo y paredes.

FIGURE 7.1: Example cutting plan from a shipbreaking yard prepared by a worker and foreperson to disassemble a large steel structure. © IEEE

vessels prefer the affordability of low-cost labor in developing countries wherein the competitive landscape prioritizes employment and profit over safety and regulation. The automation of such disassembly operations can help reduce the exposure to risks and hazards by improving occupational safety and decreasing the dependence on low-cost labor for dangerous tasks. For instance, these productivity and safety benefits are explored in robot-assisted building deconstruction [181].

This chapter is motivated by the need to break down large structures using gas torches into smaller units as seen in shipbreaking and metal scrapyards. In these unstructured and hazardous environments, an incorrect sequence of cuts can lead to a variety of dangers such as: fragments falling on the worker; the structure tipping over due to a shifting center of gravity; the structure collapsing on the worker, among other risks. As such, workers and forepersons agree on a cutting plan prior to conducting any cuts. These plans are primarily derived from the experience and intuition of the skilled workers and forepersons as well as general safety guidelines (see Fig. 7.1).

Nevertheless, safety planning can be improved by reducing the dependence on subjective assessment and intuition, thereby empowering workers and safety teams with automated sequencing and computational evaluation techniques. In effect, by utilizing physics-based simulations and safety modeling, the algorithmic generation of cutting sequences would enable more concrete evaluation of cutting scenarios. Thus, cutting sequences can be generated based on physical laws and can then be refined before execution through the domain knowledge and experience of workers and forepersons. We believe that our automated decision scheme is a step in the direction of improving the safety planning of structural disassembly.

In this chapter, we tackle the sequential decision problem of selecting the order of cuts on input structures of varied shapes and sizes. We assume that the cut locations are determined *a priori* and provided to our algorithm, which then sequences these cuts for a safe operation (Fig. 7.2). Here, safety is related to protecting the cutter (a human worker or a robot) from hazards by keeping them as far away from falling segments.

In our formulation, the structure is modeled as a linkage where its segments are considered links connected to each other via (rigid) joints. Here, the action of cutting is to disconnect two links at a particular joint. Our algorithm gives a series of

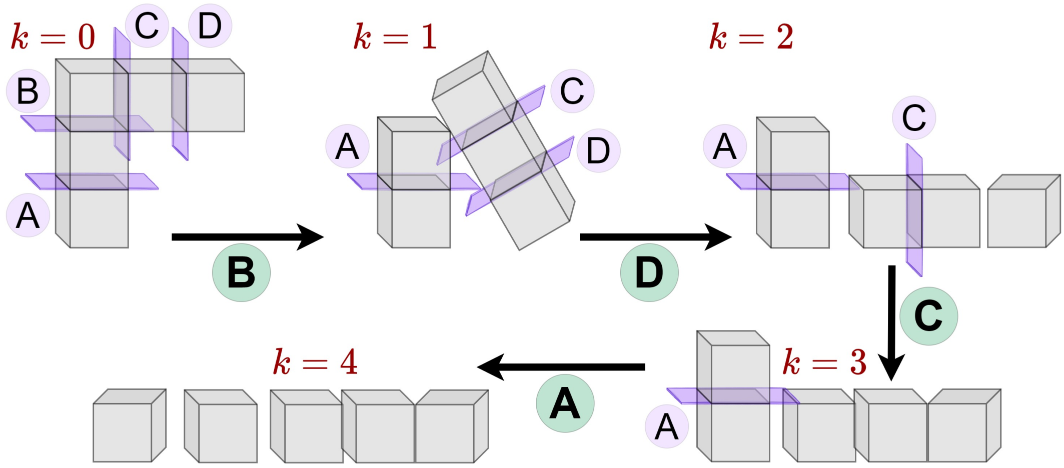


FIGURE 7.2: Conceptual diagram illustrating the sequential decision problem of structural disassembly. The input is a partitioned object with segments and cutting locations. At each step k, the decision agent selects a cut to execute until the structure is fully-disassembled. Each choice impacts the safety of the environment whereby the resulting sequence $(\mathsf{B}, \mathsf{D}, \mathsf{C}, \mathsf{A})$ can be assessed. The goal is to find a sequence that maximizes overall safety in the environment, *i.e.*, the cutter and their surroundings. © IEEE

decisions each of which is selecting one of these joints and disconnecting it. The cuts are performed in physics-based simulations wherein the safeties of the decision outcomes are computed to provide feedback to the decision agent. The number of decisions (chosen cuts) is thus equal to the number of joints n, resulting in n steps to completely fragment the object into its constituent segments. The goal is then to choose a sequence of cuts that fully disassembles the structure while maximizing safety. Here, safety considers dynamic factors which concern the structure's physics as it falls in the environment, as well as geometric factors which concern the proximity of moving segments (*i.e.*, hazards) to the cutter's location. These constructs are precisely defined in the mathematical model developed in Section 7.4.

This decision problem can be stated as a combinatorial optimization, where the objective is to find the sequence maximizing safety from a finite and discrete set of cutting sequences. Here, exhaustive search scales factorially with the input size n (the number of choices, *i.e.*, joints) becoming intractable for larger inputs. As such, we develop a greedy decision search to prune large parts of the search space and satisfice the problem's goal.

The core novel contributions of this chapter are:

- Devising an algorithm for the sequencing of cuts on partitioned structures for safely disassembling them.
- Formulating a mathematical model for safety that incorporates both dynamic and geometric factors.
- Evaluating the decision agent's performance and runtime in simulation against diverse test objects.

We emphasize our assumption that the input structure is partitioned *a priori* into constituent segments. This can be accomplished by either manually partitioning the object (in practice, based on the experience of workers and safety forepersons as in Fig. 7.1), or by utilizing a volumetric partitioning algorithm such as [182] which is skeleton-based, [183] which is search-based, or [184] which is convexity-based. We instead focus on developing a decision scheme for algorithmically disassembling such partitioned structures while maximizing safety for the environment and the cutter. In turn, the cutter's ideal location is computationally prescribed at each step depending on the state and on the cutting tool's reach. We elaborate the specifics of our modeling and procedures in Sections 7.3–7.5.

7.2 Related Work

In this section, we review existing work in sequencing algorithms and approaches for product disassembly as well as planning methods for building deconstruction.

7.2.1 Disassembly Sequence Planning

There is extensive work in disassembly sequence planning for the end-of-life handling of industrial products. [185] reviews the recent developments of robotic applications in product disassembly, distinguishing between predefined disassembly processes and more adaptable and flexible disassembly schemes. Many optimization-based approaches are applied for the disassembly process on general product structures. For instance, [186] uses multi-objective optimization to maximize parallelism, ergonomics, workload balancing, while minimizing disassembly time and product rotation count. [187] formulates the problem as an extended AND-OR graph while considering practical constraints such as reuse probability and environmental impacts. [188] represents the problem as a precedence graph and minimizes the total disassembly cost via integer programming. Other optimization schemes incorporate human–robot collaboration [189], [190]. Elsewhere, evolutionary [191] and genetic algorithms [192], [193] are applied extensively for multi-objective optimization under constraints. Hybrid approaches are also observed such as using genetic algorithms and AND-OR graphs [194], using genetic algorithms and fuzzy logic [195], or integrating several cognitive functions with a knowledge base [196]. In addition a wide variety of Petri net representations are used to model the process and resolve it using optimization-based approaches [197], [198] or fuzzy inference [199]. In addition, more specialized methods are tailored to a particular range of products such as electronics [200]–[202].

While these approaches address sequence generation for disassembling a variety of product structures, they do not consider the transient effects of the structure's disassembly and its impacts on the safety of its surroundings. In part, this is due to the structured setting of the product's disassembly and the comparatively smaller scale of the objects (*e.g.*, consumer electronics). Instead, our algorithm targets the breakdown of large structures, which expose the cutter and surrounding environment to hazards. To the best of our knowledge, we present the first work that targets safety maximization during disassembly by considering the dynamic and geometric factors of the structure's breakdown.

7.2.2 Building Deconstruction Planning

A related area of research is the deconstruction of buildings and its planning. [203] surveys many aspects related to the life-cycle of buildings and mentions uses and examples of automated planning for deconstruction. In addition, [204] reviews key problems associated with building demolition and the opportunities for automation to mitigate their effects. In the case of planning, multi-objective optimization schemes are applied with great variety for selective disassembly planning [205] and deconstruction strategy planning [206]. For automated prefabrication, [207] presents a method comparing source and target structure configurations with known parts to sequence their disassembly and reassembly. Such deconstruction plans contrast with more lower-level task planning as in robot-assisted deconstruction [181], [208] and refurbishing [209].

In recent developments, Building Information Modeling (BIM) systems are exploited to obtain structured representations of building parameters for deconstruction planning [210]–[214] and for deconstruction waste management [215]. However, BIM representations require considerable explicit modeling and are available mostly for newer and highly-organized construction projects. In contrast, many deconstruction projects involve aging structures which lack a BIM representation. Moreover, many large structures are not buildings and are incompatible with the semantics of BIM systems. In effect, 3D imaging [216] is a flexible alternative to obtaining structure representations.

In many of these planning techniques, building deconstruction is tackled statically without considering the kinetics of falling fragments. This is due to the highly-structured and highly-regulated nature of the construction industry where it can be assumed that mechanized operations can safely execute the disassembly plans via established procedures. Additionally, in the case of BIM, disassembly plans can be generated with great precision since much of the building components and parameters would be explicitly known. In contrast, scrapyard environments are highly unstructured and hazardous requiring workers to thermally cut structures using gas torches. Here, the kinetics of structural disassembly are crucial for occupational safety. To the best of our knowledge, we present the first such algorithm to maximize safety during structural disassembly.

7.3 Problem Formulation

Our structural disassembly problem involves the following elements:

1. The environment (3D space) containing the partitioned input structure, the cutter's position, and the available decisions, *i.e.*, the remaining cutting locations on the structure;
2. The dynamics of the environment given the chosen decisions; and
3. The safety function to score decisions.

We recall that the structure is partitioned into n_{links} segments. Its topology can be represented as a kinematic chain with n_{links} links (the segments) and n joints (the cutting locations). In addition, the physical properties of each link are known (and configured in simulation).

Formally, the environment state at decision step k is s_k which describes the object's state and the cutter's position $\mathbf{c}_k \in \mathbb{R}^3$. Here, the object's state encodes the geometric and dynamic information of each structure segment at step k. This would include, for instance, each segment's shape, size, pose, mass, and so on. The state s_k thus describes at step k the properties of each segment and the cutter's location $\mathbf{c}_k$. A decision is to select and disconnect one of the structure's joints. We denote the decision at step k as $a_k \in \mathcal{A}_k$. Here, $\mathcal{A}_k$ is the set of available decisions, *i.e.*, the joints that have yet to be disconnected. Accordingly, a decision sequence can be expressed as $\mathbf{a} = (a_k)_{k=0}^{n-1} = (a_0, \ldots, a_{n-1})$. Note that the decision sequence bijectively maps the decision steps k to the joint indices j, that is, $\mathbf{a} : \{k\}_{k=0}^{n-1} \leftrightarrow \{j\}_{j=0}^{n-1}$.

The dynamics of the environment can thus be expressed as $s_{k+1} = f(s_k, a_k)$ where f maps to the next state s_{k+1} given the decision a_k taken at the current state s_k. In our case, f is a physics-based simulation where selecting a cut (decision a_k) at a particular scene (current state s_k) leads to motions in the scene. Once all entities in the simulation cease to move, the next state s_{k+1} (*i.e.*, the resultant scene) is realized. This is in agreement with safety practices in the scrapyard, where cuts are performed on the structure after it stabilizes and its pieces cease to move. Now, the safety of a decision can be measured using a safety function $S : \mathcal{A}_k \rightarrow [0, 1]$ that maps from decisions to the unit interval. Specifically, the safety function scores the decision a_k based on its outcome and transition to s_{k+1}, *i.e.*, $S(a_k)$ is a function of s_{k+1} and the transition from s_k to s_{k+1}. We now define the safety of a decision sequence $\mathbf{a}$ as the weighted geometric mean of its individual decision safeties $S(a_k)$, as in:

$$S(\mathbf{a}) = \prod_{k=0}^{n-1} S(a_k)^{\lambda_k} = S(a_0)^{\lambda_0} \cdots S(a_{n-1})^{\lambda_{n-1}} \tag{7.1}$$

The weights λ_k are expressed as a proportion, *i.e.*, $\lambda_k \in [0, 1]$ and $\sum_{k=0}^{n-1} \lambda_k = 1$. These weights λ_k are assigned such that the worst-case decision $a_{\mathsf{wc}} = \operatorname{argmin}_{a_k \in \mathbf{a}} S(a_k)$ has a corresponding weight of λ_{wc} whereas all remaining weights $\lambda_{k \neq \mathsf{wc}} = \frac{1-\lambda_{\mathsf{wc}}}{n-1}$ are uniform. By setting $\lambda_{\mathsf{wc}} = \frac{1}{2}$, the sequence safety is penalized by the worst-case decision irrespective of the sequence length. In doing so, the sequence safety can assess and compare sequences of any length yet remains skewed by its most dangerous decision. By design, this multiplicative formulation heavily penalizes the sequence safety for unsafe decisions. For instance, one strictly unsafe decision with $S(a_k) = 0$ would result in a strictly unsafe sequence $S(\mathbf{a}) = 0$. This is a desirable modeling choice due to the sequential nature of the problem (non-episodicness) given the permanent harm inflicted by potential hazards. Note that λ_{wc} quantifies the weight of the worst-case decision, meaning that a higher λ_{wc} leads to a more conservative measure of sequence safety.

Finally, the goal is to maximize the sequence's safety:

$$\mathbf{a}^* = \operatorname{argmax}_{\mathbf{a}} S(\mathbf{a}) = \operatorname*{argmax}_{a_0, \ldots, a_{n-1}} \prod_{k=0}^{n-1} S(a_k)^{\lambda_k} \tag{7.2}$$

In the next sections, we define our safety model S as well as our decision algorithm to determine the cutter's position $\mathbf{c}_k$ and the decisions a_k.

7.4 Safety Model

The safety $S(a_k)$ of a decision a_k is modeled in such a way to capture the dynamic and geometric outcomes of the transition from s_k to s_{k+1}, This starts when the cut is made and stops when all segments in the scene stop moving.

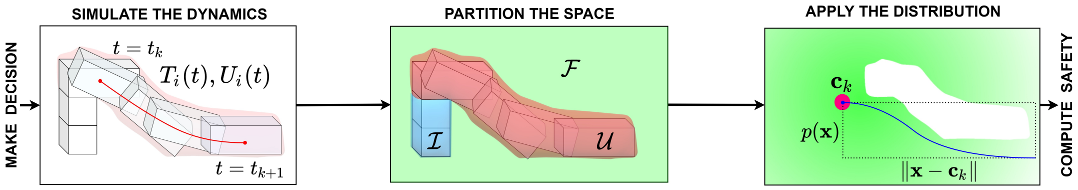


FIGURE 7.3: Illustrating the computational elements of the safety model. A decision a_k is simulated wherein motions begin at t_k and end at t_{k+1}. The environment space $\mathbb{R}^3$ is partitioned into unsafe ($\mathcal{U}$), immobile ($\mathcal{I}$), and free ($\mathcal{F}$). The safety model considers both dynamic effects (aggregate motion of the structure) and geometric effects (proximity of moving segments to the cutter position $\mathbf{c}_k$). The cutter's ideal position $\mathbf{c}_k$ is computed algorithmically. © IEEE

We define the safety $S(a_k)$ as:

$$S(a_k) = S_d(a_k)\, S_g(a_k) \tag{7.3}$$

where $S_d(a_k)$ is the dynamic safety and $S_g(a_k)$ is the geometric safety. We note that each of S, S_d, and S_g map decisions to the unit interval, *i.e.*, $\mathcal{A}_k \to [0, 1]$.

By design, S_d captures the kinetic outcomes of cutting the structure, *i.e.*, the 'intensity' of the segments' motion. In contrast, S_g captures the kinematic outcomes of cutting the structure, *i.e.*, the 'closeness' of the segments' paths as they fall relative to the cutter's position. These modeling choices are based on the following assertions for decision safety:

- The decision is safer when the aggregate motion of the segments in the scene has a lower magnitude.
- The decision is safer when the aggregate traversal of the segments in the scene is further from the cutter's position.

Both S_d and S_g are formally defined below accordingly.

7.4.1 Dynamic Safety

While there are many ways to express the magnitude of the segments' aggregate motion, we wish to concisely capture inertial effects during motion. As such, a straightforward choice is to use the kinetic energies of each segment.

We first define the time interval $t \in [t_k, t_{k+1}]$ wherein the transition from s_k to s_{k+1} takes places and the segments start ($t = t_k$) and stop moving ($t = t_{k+1}$). Now, we define $T_i(t)$ and $U_i(t)$ to be the kinetic and gravitational potential energies of each segment $i \in \{1, n_{\text{links}}\}$ as well as their totals $T(t) = \sum_i T_i(t)$ and $U(t) = \sum_i U_i(t)$. We note that the total mechanical energy $T(t) + U(t)$ in the scene is conserved for $t \in [t_k, t_{k+1}]$ since dissipative forces (*e.g.*, friction, air resistance) are negligible. As such, we can define the normalized energies $\tilde{T}(t) = \frac{T(t)}{T(t)+U(t)}$ and $\tilde{U}(t) = \frac{U(t)}{T(t)+U(t)}$. These capture the ratios of kinetic and potential energies respectively. Stated differently, they capture the instantaneous ratio of mechanical energy due to motion and to non-motion.

We would like S_d to increase with a larger ratio due to non-motion, *i.e.*, when the structure is disassembled more gently. We thus define dynamic safety as follows:

$$S_d(a_k) = \min_{t \in [t_k, t_{k+1}]} \tilde{U}(t) \tag{7.4}$$

As S_d is the minimum of $\tilde{U}(t)$ across the transition from s_k to s_{k+1}, Eq. (7.4) yields the desired properties of mapping to $[0, 1]$ and of decreasing with higher aggregate motion. The relationship between the energies ($\tilde{U}$, $\tilde{T}$) and the decision a_k is implicit whereby the decision's outcome determines the energy values through the aforementioned manner.

7.4.2 Geometric Safety

In addition to dynamic safety, we wish to consider the proximity of the segments' traversal relative to the cutter's location during the transition interval $t \in [t_k, t_{k+1}]$.

For this, we partition the environment space $\mathbb{R}^3$ into three sets (Fig. 7.3): the unsafe space $\mathcal{U}$, the immobile space $\mathcal{I}$, and the free space $\mathcal{F}$. These three spaces are defined as follows.

The unsafe space $\mathcal{U}$ contains all points visited by moving object segments during the transition interval $t \in [t_k, t_{k+1}]$ after a decision a_k. This can be expressed as $\mathcal{U} = \bigcup_{t \in [t_k, t_{k-1}]} V(t)$ where $V(t) = \bigcup_{i=1} V_i(t)$ is the set of points contained within all moving segments at time t and $V_i(t)$ is the set of points contained within the moving segment i at time t. Thus, $\mathcal{U}$ represents the collision set in the environment containing all points visited by moving segments throughout their motions.

The immobile space $\mathcal{I}$ contains the points of all segments that remain immobile throughout the transition interval $t \in [t_k, t_{k+1}]$ after a decision a_k. These are the stationary segments, which are not hazards. The free space $\mathcal{F}$ contains the points that remain unoccupied throughout the transition interval $t \in [t_k, t_{k+1}]$ after a decision a_k. These are safe and empty locations that have not experienced collisions.

Using these definitions, we distinguish the unsafe space $\mathcal{U}$ from the safe space $\mathcal{S} = \mathcal{U}^{\complement} = \mathcal{F} \cup \mathcal{I}$. As such, the indicator function $\mathbf{1}_{\mathcal{S}} : \mathbb{R}^3 \to \{0, 1\}$ partitions the environment into unsafe positions where $\mathbf{1}_{\mathcal{S}}(\mathbf{x}) = 0$ and safe positions where $\mathbf{1}_{\mathcal{S}}(\mathbf{x}) = 1$. This safety indicator function expresses the safety of a position in the environment.

We algorithmically determine an ideal position $\mathbf{c}_k$ for the cutter as described in Section 7.5. This $\mathbf{c}_k$ represents the prescribed location that the cutter should cut from. In actuality, the cutter may stray from their ideal position and we thus represent their position as a random variable $\mathbf{x} \in \mathbb{R}^3$. We model $\mathbf{x}$ to follow a unimodal and symmetric distribution centered at its mode (the cutter's ideal position $\mathbf{c}_k$) and with radial decay. The intent is to decrease the probability of the cutter's presence as $\|\mathbf{x} - \mathbf{c}_k\|_2$ grows. For this, a trivariate Gaussian distribution with mean $\mathbf{c}_k$ and covariance matrix $r^3 \mathrm{I}_3$ is appropriate. With $\mathbf{x} \sim \mathcal{N}(\mathbf{c}_k, r^3 \mathrm{I}_3)$, we obtain its probability density function $p(\mathbf{x})$ by evaluating the multinormal PDF with mean $\mathbf{c}_k$ and covariance $r^3 \mathrm{I}_3$.

$$p(\mathbf{x}) = \frac{1}{r^3 \sqrt{(2\pi)^3}} \exp\left[-\frac{1}{2r^2} \|\mathbf{x} - \mathbf{c}_k\|_2^2 \right] \tag{7.5}$$

Finally, define the geometric safety $S_g(a_k)$ as:

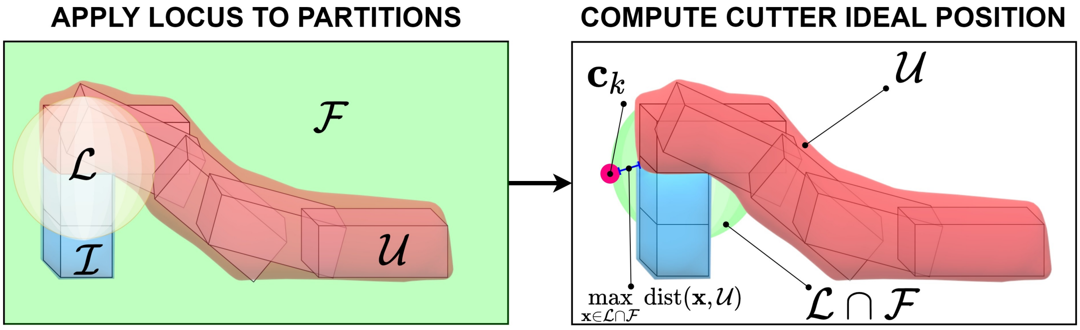

APPLY LOCUS TO PARTITIONS
$\mathcal{F}$
$\mathcal{L}$
$\mathcal{I}$
$\mathcal{U}$
COMPUTE CUTTER IDEAL POSITION
$\mathbf{c}_k$
$\mathcal{U}$
$\max_{\mathbf{x}\in\mathcal{L}\cap\mathcal{F}} \mathrm{dist}(\mathbf{x},\mathcal{U})$
$\mathcal{L}\cap\mathcal{F}$

FIGURE 7.4: The cutter's ideal position is computed as the locus point in the free space ($\mathbf{x} \in \mathcal{L} \cap \mathcal{F}$) that is maximally-distant from the unsafe space $\mathcal{U}$. © IEEE

$$S_g(a_k) = \mathbb{E}[\mathbf{1}_{\mathcal{S}}(\mathbf{x})] = \int_{\mathbf{x}\in\mathbb{R}^3} \mathbf{1}_{\mathcal{S}}(\mathbf{x})\, p(\mathbf{x})\, \mathrm{d}\mathbf{x} \tag{7.6}$$

In this manner, $S_g(a_k)$ represents the cutter's expected safety in the environment after decision a_k given their ideal position $\mathbf{c}_k$. Also since $\mathbf{1}_{\mathcal{S}}(\mathbf{x}) \in \{0, 1\}$, S_g maps to $[0, 1]$ as desired. Moreover, the parameter r can be interpreted as a 'radius' wherein the cutter's safety is most crucial. A smaller r decays the safety scores rapidly prioritizing the safety nearest to the prescribed position. Conversely, a larger r decays the safety scores more slowly giving added consideration to regions further away from the prescribed position.

In implementation, the formulation is discretized as follows: the segments' kinematic and kinetic information are sampled from the simulation as time-series data; the set $\mathcal{S}$ and its indicator function $\mathbf{1}_{\mathcal{S}}$ are implemented as a binary voxel grid; and the Eqs. (7.4) and (7.6) are approximated using discrete sums.

7.5 Decision Algorithm

With the safety function S defined and the decision goal $(\mathbf{a}^*, S^*)$ established in Eq. (7.2), we now describe the procedures and schemes used to solve the decision problem.

7.5.1 Cutter's Ideal Position

The decision agent not only generates a disassembly sequence, but also specifies the cutter's prescribed position for each cut (Fig. 7.4). In effect, the safety model requires this position $\mathbf{c}_k$ as seen in Eq. (7.5). We emphasize the advantage of simulating the environment dynamics: the ability to observe the cut's outcome first and then decide where the best position to cut from should have been. In this way, the cutter's ideal position is determined *a posteriori*. Specifically, the decision agent attempts and simulates a cut a_k, measures its outcome s_{k+1}, and then determines $\mathbf{c}_k$, *i.e.*, the safest position to cut from. Here, we specify a procedure to compute $\mathbf{c}_k$ given the decision a_k and its outcome s_{k+1}.

```
define ComputeIdealPosition($a_k$, $s_{k+1}$, $m$):
    $(\mathcal{F}, \mathcal{I}, \mathcal{U}) \leftarrow$ GetPartitions($s_{k+1}$)
    $\mathcal{L} \leftarrow$ GenerateLocus($a_k$, $m$)
    $\mathbf{c}_k \leftarrow \operatorname*{argmax}_{\mathbf{x} \in \mathcal{L} \cap \mathcal{F}} \text{dist}(\mathbf{x}, \mathcal{U})$
    return $\mathbf{c}_k$
```

After the cut, we retrieve from s_{k+1} the space's partitioning into unsafe ($\mathcal{U}$), immobile ($\mathcal{I}$), and free ($\mathcal{F}$) as defined in Section 7.4.2. The cutter's position locus $\mathcal{L}$ contains all candidate ideal positions and depends on the cutting margin $m > 0$ defined by the cutting tool's length.

For instance, the cutting torch used in shipbreaking is around 2 m long, giving the cutter a margin of $m = 2$ m. We model the cutter's locus as a sphere centered at the joint location of a_k with radius m. We wish to position the cutter in the free space furthest from the unsafe space. This is expressed as the point in $\mathcal{L} \cap \mathcal{F}$ with maximum distance from $\mathcal{U}$, or $\text{argmax}_{\mathbf{x} \in \mathcal{L} \cap \mathcal{F}} \text{dist}(\mathbf{x}, \mathcal{U})$.

We do not consider the case of segments so large that $\mathcal{L} \cap \mathcal{F} = \varnothing$ wherein the cutter cannot safely reach the joint within a radius m. In such cases, the object must be re-partitioned in a reasonable manner to enable safer cutting. In a discretized implementation, one can rank the candidates by distance and select the furthest that is feasible in terms of cutting ergonomics. Now, a decision a_k can be evaluated as follows.

```
define EvaluateDecision($a_k$):
    $s_{k+1} \leftarrow$ SimulateDecision($a_k$)
    $\mathbf{c}_k \leftarrow$ ComputeIdealPosition($a_k$, $s_{k+1}$, $m$)
    $S_k \leftarrow$ ComputeSafety($a_k$, $\mathbf{c}_k$, $r$)
    return $(S_k, \mathbf{c}_k)$
```

The procedure SimulateDecision represents the physics simulator which loads the environment containing the input paritioned model and computes the outcomes of the cut a_k. The procedure ComputeSafety implements Eqs. (7.3)–(7.6).

7.5.2 Decision Search Schemes

We develop three alternative searches: exhaustive (ES), greedy (GS), and random (RS). For each search, the input is the partitioned object with n joints and the output is the solution $(\mathbf{a}, S, \mathbf{C})$ containing the decision sequence $\mathbf{a}$, its safety S, and the sequence $\mathbf{C}$ of ideal positions. In addition the parameters r (see Section 7.4.2) and m (see Section 7.5.1) are specified based on application requirements.

ES finds the optimal solution $\mathbf{a}^*$ of Eq. (7.2). Both ES and RS provide a basis of comparison for computational complexity and safety performance. GS approximates $\mathbf{a}^*$ by optimizing locally $a_k = \text{argmax}_{a \in \mathcal{A}_k} S(a_k)$. The schemes are specified in Algorithms 4–6. We note that $\mathcal{A}$ is the set of all decision sequences. These are permutations (without replacement) of the structure's n joints yielding a cardinality of $n!$ sequences. We recall that $\mathcal{A}_k$ is the set of remaining decisions available at step k. The cardinality of $\mathcal{A}_k$ is $n - k$ and its contents depend on $\mathcal{A}_{k-1}$ and a_{k-1} (for $k > 0$).

Algorithm 4: Exhaustive Decision Search

```
Input      : Partitioned structure with $n$ joints.
Output     : Decision sequence $\mathbf{a}$, its safety $S$, and cutter ideal positions $\mathbf{C}$.
Parameters : Safety radius $r$ and cutting margin $m$.
Initialize safety for the solution sequence: $S \leftarrow 0$
Iterate over all possible decision sequences.
for all $\mathbf{a}_{\mathsf{cand}} \in \mathcal{A}$ do
    Iterate over each decision per sequence.
    for each $a_k \in \mathbf{a}_{\mathsf{cand}}$ do
        $(S_k, \mathbf{c}_k) \leftarrow$ EvaluateDecision($a_k$)
    Compute the candidate solution components.
        $S_{\mathsf{cand}} \leftarrow \prod_{k=0}^{n-1} S_k^{\lambda_k}, \quad \mathbf{C}_{\mathsf{cand}} \leftarrow (\mathbf{c}_0, \dots, \mathbf{c}_{n-1})$
    Accept candidate solution upon improvement.
    if $S_{\mathsf{cand}} > S$ then
        $(\mathbf{a}, S, \mathbf{C}) \leftarrow (\mathbf{a}_{\mathsf{cand}}, S_{\mathsf{cand}}, \mathbf{C}_{\mathsf{cand}})$
return $(\mathbf{a}, S, \mathbf{C})$
```

7.5.3 Asymptotic Performance

We express the computational complexity as the total decision evaluations $\mathsf{C}(n)$ for an input size n (number of decisions). A decision evaluation refers to a call to the `EvaluateDecision` routine which involves simulating the decision, computing its cutter's ideal position, and computing its safety. ES as indicated in Algorithm 4 performs $n \times n!$ evaluations. Even with efficient implementation, *e.g.*, memoization (storing reusable states), these reduce to $\mathsf{C}(n) = \sum_{k=0}^{n-1} \frac{n!}{(n-k-1)!} \in \mathcal{O}(n!)$ evaluations. GS runs $\mathsf{C}(n) = \sum_{k=0}^{n-1}(n-k) = \frac{n(n+1)}{2} \in \mathcal{O}(n^2)$ evaluations and RS runs $\mathsf{C}(n) = n \in \mathcal{O}(n)$ evaluations.

7.6 Evaluation in Simulation

We implement our decision environment using the simulator Gazebo [98] to evaluate our decision agent's performance using each of the search schemes against different inputs.

7.6.1 Simulation Environment

In our simulations, the object segments are considered rigid bodies connected as a kinematic chain. A cut is implemented as an instantaneous disconnection between two segments. Our simulations model rigid-body dynamics, gravitational and frictional forces, and collisions. A simulated experiment consists of loading the input object, executing cuts per the decision agent, and updating the environment's state until a solution for the object's full disassembly is chosen. We test our implementation on the 12 partitioned objects (Fig. 7.5). We illustrate the decision outputs of each decision scheme against the object `5B` in Table 7.1 and in Fig. 7.6.

7.6.2 Simulation Results

As can be seen in Fig. 7.6, ES and GS yield similar decisions for object `5B` differing only in their second and third decisions which are permuted. In contrast, RS can yield

Algorithm 5: Greedy Decision Search

```
Input      : Partitioned structure with $n$ joints.
Output     : Decision sequence $\mathbf{a}$, its safety $S$, and cutter ideal positions $\mathbf{C}$.
Parameters : Safety radius $r$ and cutting margin $m$.
Iterate over each decision step of the solution.
for $k \leftarrow 0, \dots, n-1$ do
    Initialize safety for the current decision: $S_k \leftarrow 0$
    Iterate over all currently available decisions.
    for all $a_{\mathsf{cand}} \in \mathcal{A}_k$ do
        $(S_{\mathsf{cand}}, \mathbf{c}_{\mathsf{cand}}) \leftarrow$ EvaluateDecision$(a_{\mathsf{cand}})$
        Accept candidate decision upon improvement.
        if $S_{\mathsf{cand}} > S_k$ then
            $(a_k, S_k, \mathbf{c}_k) \leftarrow (a_{\mathsf{cand}}, S_{\mathsf{cand}}, \mathbf{c}_{\mathsf{cand}})$
Compute the solution's components.
  $\mathbf{a} \leftarrow (a_0, \dots, a_{n-1})$,    $S \leftarrow \prod_{k=0}^{n-1} S_k^{\lambda_k}$
  $\mathbf{C} \leftarrow (\mathbf{c}_0, \dots, \mathbf{c}_{n-1})$
return $(\mathbf{a}, S, \mathbf{C})$
```

Algorithm 6: Random Decision Search

```
Input      : Partitioned structure with $n$ joints.
Output     : Decision sequence $\mathbf{a}$, its safety $S$, and cutter ideal positions $\mathbf{C}$.
Parameters : Safety radius $r$ and cutting margin $m$.
Iterate over each decision step of the solution.
for $k \leftarrow 0, \dots, n-1$ do
    Randomly select a currently available decision.
      $a_k \leftarrow$ rand $\mathcal{A}_k$
    $(S_k, \mathbf{c}_k) \leftarrow$ EvaluateDecision$(a_k)$
Compute the solution's components.
  $\mathbf{a} \leftarrow (a_0, \dots, a_{n-1})$,    $S \leftarrow \prod_{k=0}^{n-1} S_k^{\lambda_k}$
  $\mathbf{C} \leftarrow (\mathbf{c}_0, \dots, \mathbf{c}_{n-1})$
return $(\mathbf{a}, S, \mathbf{C})$
```

highly-dangerous decisions where large chunks of the structure tumble dynamically. In effect, the safeties of its worst decisions are $S_1 = 0.28$ and $S_3 = 0.48$.

More generally, we summarize in Table 7.2 the decision agent's performance for each search scheme against each test structure. We note the safety of each scheme's solution $S(\mathbf{a})$ and its search cost $\mathsf{C}(n)$ measured as the number of decision evaluations (see Section 7.5.3). For example, ES computes for object 3A the optimal sequence $\mathbf{a}^*$ with safety $S(\mathbf{a}^*) = 0.7116$ using $\mathsf{C}(3) = 18$ decision evaluations. Similarly, GS computes the greedy solution $\mathbf{a}^\dagger$ for object 3A and its safety $S(\mathbf{a}^\dagger) = 0.6989$. The random solution is uniformly-sampled, and therefore we compute its expected safety and worst-case safety, which for object 3A are $\mathbb{E}[S(\mathbf{a})] = 0.6776$ and $S(\mathbf{a}_{\mathsf{wc}}) = 0.6495$ where $\mathbf{a}_{\mathsf{wc}} = \operatorname{argmin}_{\mathbf{a} \in \mathcal{A}} S(\mathbf{a})$.

We simulate all $n!$ sequences for each n-segment structure in order to compute the safety $S(\mathbf{a}^*)$ of the optimal sequence, the expected safety $\mathbb{E}[S(\mathbf{a})]$ of the uniformly sampled sequence, and the safety $S(\mathbf{a}_{\mathsf{wc}})$ of the worst-case sequence.

TABLE 7.1: Outputs for object 5B showing the sequence and its safeties © IEEE

Scheme	**Sequence a**	**Decision Safeties S_k**
Exhaustive	(4, 2, 3, 5, 1)	(0.79, 0.76, 0.78, 0.63, 0.65)
Greedy	(4, 3, 2, 5, 1)	(0.79, 0.76, 0.71, 0.62, 0.64)
Random	(2, 1, 5, 3, 4)	(0.79, 0.28, 0.74, 0.48, 0.99)

TABLE 7.2: Simulation results of each scheme's cost and performance © IEEE

Input	**Exhaustive**		**Greedy**		**Random**		
(Obj, n)	C(n)	$S(\mathbf{a}^*)$	C(n)	$S(\mathbf{a}^\dagger)$	C(n)	$\mathbb{E}[S(\mathbf{a})]$	$S(\mathbf{a}_{\text{wc}})$
(3A, 3)	18	0.7116	6	0.6989	3	0.6776	0.6495
(3B, 3)	18	0.8402	6	0.8378	3	0.8150	0.7505
(3C, 3)	18	0.8104	6	0.8097	3	0.7561	0.7149
(3D, 3)	18	0.7628	6	0.7270	3	0.7180	0.6470
(4A, 4)	96	0.9226	10	0.9050	4	0.9045	0.8950
(4B, 4)	96	0.8053	10	0.7980	4	0.7268	0.6480
(4C, 4)	96	0.7934	10	0.7858	4	0.7402	0.5661
(4D, 4)	96	0.8053	10	0.7929	4	0.7797	0.6701
(5A, 5)	600	0.8571	15	0.8467	5	0.7365	0.5259
(5B, 5)	600	0.6849	15	0.6838	5	0.5599	0.4538
(5C, 5)	600	0.8068	15	0.7367	5	0.6652	0.5322
(5D, 5)	600	0.8602	15	0.7861	5	0.7680	0.6990

7.6.3 Discussion of the Results

We directly compare in Table 7.3 the performance of GS against that of ES and RS. We compare the greedy solution against the optimal solution via the ratio of their safeties $\frac{S(\mathbf{a}^\dagger)}{S(\mathbf{a}^*)}$. Furthermore, we compute the relative improvement of the greedy solution's safety against that of the random solution's expected and worst-case safeties. For our 12 test objects, the greedy solution $\mathbf{a}^\dagger$ yields a safety that is on average 96.72% of the optimal sequence's safety $S(\mathbf{a}^*)$. In some cases, the greedy solution is near optimal where $\frac{S(\mathbf{a}^\dagger)}{S(\mathbf{a}^*)} > 99\%$ such as with objects $\{$3B, 3C, 4B, 4C, 5B$\}$. Moreover, the greedy solution $\mathbf{a}^\dagger$ performs on average 6.10% better than uniformly-sampled random searching and 17.83% better than the worst-case sequence, the latter of which would yield potential risks and hazards. In addition, our results reveal that some objects are inherently safer to cut where decisions weakly impact the safety outcomes, *e.g.*, for object 4A the solution $\mathbf{a}^\dagger$ is only 0.05% safer than random and 1.10% safer than the worst-case sequence. Conversely, some objects are inherently much more dangerous to cut (4C, 5A, 5B, 5C) where decisions greatly impact the safety outcomes, *e.g.*, for object 5B the solution $\mathbf{a}^\dagger$ is 18.12% safer than random and 33.64% safer than the worst-case sequence. Our results suggest that GS yields significantly safer decisions than RS, sometimes achieving near-optimal safety using only $\mathcal{O}(n^2)$ decision evaluations. For illustration with $n = 5$ as in object 5B (Fig. 7.6), the GS solution is found in 15 evaluations, instead of 600 as with the ES solution.

TABLE 7.3: Evaluation of the greedy scheme's performance © IEEE

Input	Greedy vs. Exhaustive	Greedy vs. Random	
(Obj, n)	$S(\mathbf{a}^\dagger)/S(\mathbf{a}^*)$	$\frac{S(\mathbf{a}^\dagger)-\mathbb{E}[S(\mathbf{a})]}{S(\mathbf{a}^\dagger)}$	$\frac{S(\mathbf{a}^\dagger)-S(\mathbf{a}_{\text{wc}})}{S(\mathbf{a}^\dagger)}$
(3A, 3)	98.22%	3.04%	7.07%
(3B, 3)	99.72%	2.72%	10.42%
(3C, 3)	99.91%	6.62%	11.71%
(3D, 3)	95.31%	1.24%	11.01%
(4A, 4)	98.09%	0.05%	1.10%
(4B, 4)	99.09%	8.92%	18.79%
(4C, 4)	99.05%	5.81%	27.96%
(4D, 4)	89.92%	1.67%	15.49%
(5A, 5)	98.79%	13.01%	37.89%
(5B, 5)	99.84%	18.12%	33.64%
(5C, 5)	91.32%	9.71%	27.76%
(5D, 5)	91.38%	2.30%	11.07%
Mean	96.72%	6.10%	17.83%

7.7 Conclusion

This chapter formalizes the safe disassembly of large structures as a sequential decision problem for which we develop an algorithm to maximize the safety for the cutter and the surrounding environment. By simulating a decision's outcomes, the agent partitions the environment into safe and unsafe. Thereafter, an ideal position for the cutter is prescribed and the decision's safety is measured. Safety is modeled to capture the decision's dynamic outcomes, *i.e.*, the structure's resulting kinetics, as well as the geometric outcomes, *i.e.*, the motion's proximity to the cutter's prescribed position. We design three decision search schemes to find a solution consisting of the decision sequence, its safety score, and the sequence of prescribed cutter positions. These schemes are tested in simulation against 12 partitioned objects wherein their safety performance and computational costs are measured. Our results show that the greedy decision search satisfices the problem with considerably less computational cost than exhaustive searching. In essence, our greedy algorithm: (1) scales better with input size n by computing $\mathcal{O}(n^2)$ rather than $\mathcal{O}(n!)$ decision evaluations; (2) yields good solutions (sometimes near-optimal); (3) is noticeably better than random selection (sometimes drastically); and, (4) is significantly better than the worst-case choice. In future work, it is worth incorporating more realistic dynamics and decision parameters to reduce the sim-to-real gap.

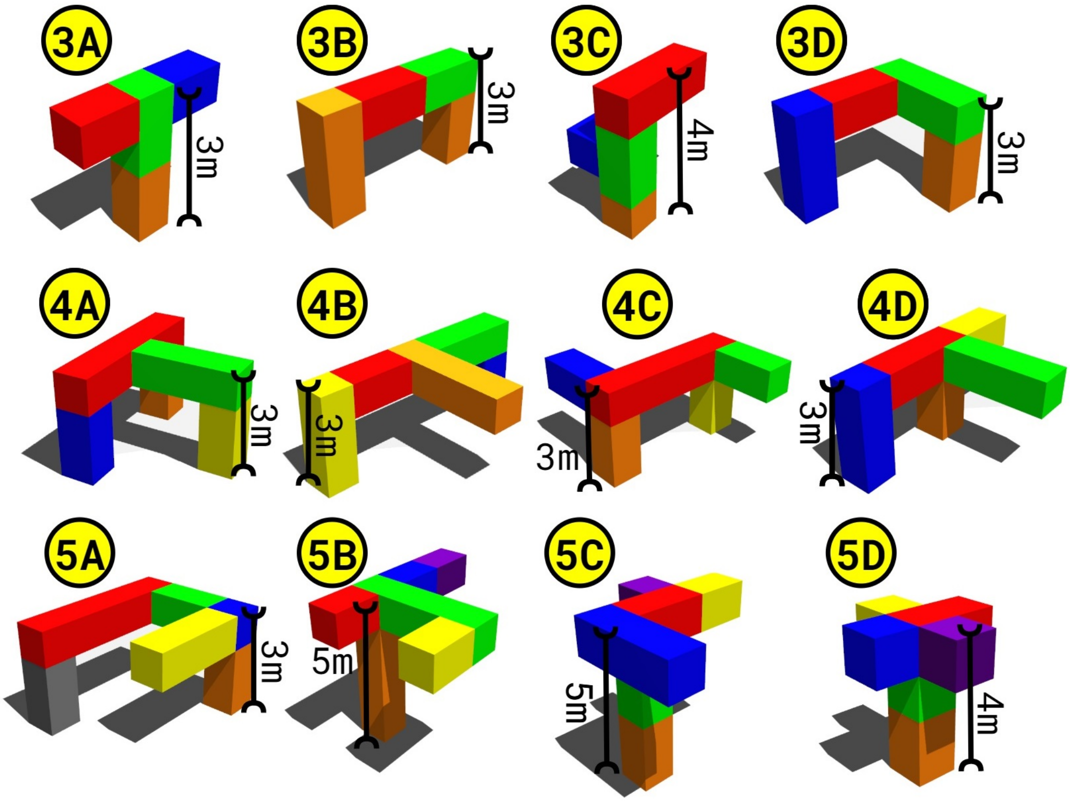


FIGURE 7.5: The 12 partitioned test objects are shown with their label indicating the number of cuts (*e.g.*, `3A` has $n = 3$) and their structural height. © IEEE

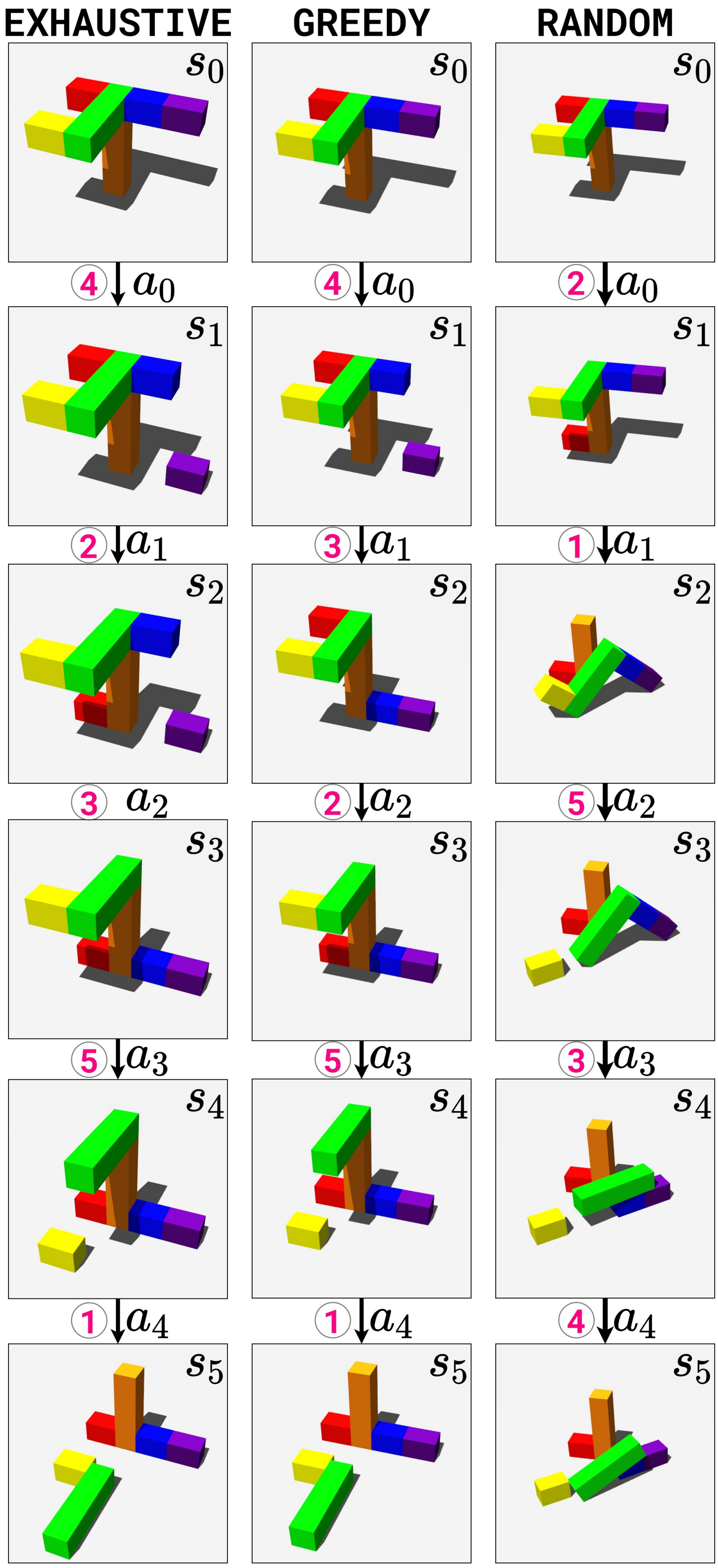


FIGURE 7.6: The decision outcomes of each search scheme is demonstrated on object 5B. We note the difference in safety outcomes between the random scheme and the other schemes. The safety scores can be found in Table 7.1. © IEEE

Chapter 8

Broader Impacts

In developing the components (Chapters 4–7) of our proposed framework (Chapter 3), we endow robotic systems with the autonomy to perform some of the operations within metal scrap recycling. Accordingly, our research would empower future developments in automated systems for metal scrapping and related applications. As of our treatment, this technology is in its research stage wherein the emphasis is on fundamentals and on the development of basic techniques—all of which can be employed towards the future implementation of engineered systems. As this work is nascent, it is non-trivial to reliably unveil ethical issues deriving from its application in larger socio-economic contexts. In performing and providing an anticipatory ethical analysis [217], we highlight the potential benefits, concerns, and risks that may result from the advancement, commercialization, and deployment of this research. While such projections may be speculative, we nevertheless inform ethical and policy considerations towards the promotion of morally desirable uses of autonomous and intelligent systems.

Keywords: *Automated metal scrap recycling, future of work, technological unemployment, autonomous and intelligent systems, robot ethics, applied ethics, anticipatory ethics.*

8.1 Beneficial Impacts

In forecasting the outcomes of our research, there are substantial beneficial impacts that may arise. In effect, the introduction of a human–robot collaborative workflow into scrapyards could eventually reduce many of the industry's inherent problems as mentioned in Chapter 2. Below, we enumerate and elaborate some salient benefits.

Improved bridging of basic and applied research in robotics.

The development of our proposed framework draws from a wide reach of research domains within robotics and intelligent agents. In retrospect, the motivation of our framework stems from applied research for domain-specific objectives within metal scrap recycling. However, the adversity of the scrapyard goes beyond application and invites the design of novel and fundamental techniques tailored to the unique challenges of the target environment and its operations.

In doing so, our research adapts, utilizes, and contributes to fundamental areas of research—*e.g.*, active perception, vision-based control, learning-based process monitoring, sequential disassembly—while simultaneously enriching applied contexts such as environmental robotics, robotic recycling, industrial robotics, robotic metal cutting, and field robotics. Through this, research societies benefit from the cross-pollination of ideas and the increased exposure and diversification of techniques.

In this manner, basic researchers would be exposed to novel challenges and problem structures inspired from and motivated by the aforementioned applied contexts. Conversely, applied researchers could extend or specialize existing fundamental techniques by adapting and tailoring them to their targeted domains.

Increased partnerships between academia and industry.

Our research targets functionalities that would address problems encountered in industrial contexts. In effect, applied research that overlaps with the interests and needs of specific industrial sectors would attract opportunities for collaboration and partnership. For illustration, our own work stems from and builds upon a collaboration between academia (our research group, the Manipulation & Environmental Robotics Lab) and industry (our corporate partner, European Metal Recycling Ltd.).

Such partnerships can lead to mutually beneficial outcomes wherein academic research and industrial practice intersect and provide their respective strengths. In particular, research groups can provide research output in the form of publications and patents. Aside from intellectual property, researchers can lower the risk of investment by demonstrating the fundamentals of the technology in the lab at an early stage. In doing so, the industrial partner can better manage financial risk in their investments in novel and emerging technologies. Conversely, the industrial partner can provide domain knowledge and empirical insights that emphasize the most important and impactful research directions. Furthermore, corporate partners provide research funding in directions deemed beneficial per their operational needs.

Furthermore, academia–industry collaboration can lead to much wider involvement downstream as the research matures. Upon successful demonstration of the fundamental techniques and early prototypes, the research efforts would mature into a development stage that may involve larger engineering teams and system integrators. Through this, the methodologies designed in the research stage can evolve at a much faster pace through various stages of technological readiness [218], from feasibility assessments and technology demonstrators leading up (upon success) to the deployment and commercialization of reliable systems. In this way, industrial participation can accelerate the attainment of the technological and economic benefits that novel methods can yield.

Improved preservation of national security.

As mentioned in Chapter 2, there is incentive for the domestic scrapping and disposal of decommissioned military vessels. This is particularly emphasized when retired naval vessels contain elements that can be weaponized. In effect, while many retired vessels are no longer economically viable to operate, they nevertheless can incorporate warship designs, weapon systems, or nuclear material. For instance, retired nuclear submarines are scrapped domestically [219] to avoid the uncontrolled circulation of nuclear material. By introducing automated metal scrapping systems, the domestic scrapping capacity in domestic shipbreaking yards would be increased. As a result, matters of national security can be preserved while the burden of maintaining inefficent naval vessels can be addressed earlier.

Increased global competitiveness of the American shipbreaking industry.

For the past two decades, at least 97% of the global tonnage in scrapped metal from shipbreaking is processed [220] in the yards of the five leading ship recycling countries—Bangladesh, China, India, Pakistan, and Turkey. In contrast, the American shipbreaking sector only accounted for around 0.3% of that global tonnage. By introducing automated metal scrapping systems alongside skilled cutting workers in American scrapyards, the operating costs of scrapping can be reduced and shipbreaking productivity overall would increase. Such improvements in cost-effectiveness and productivity would increase the global competitiveness of the domestic shipbreaking sector. Furthermore, in addressing the industry's problems mentioned in Section 2.1.3, such increased competitiveness would reduce the dependence of the domestic shipbreaking industry on U.S. government.

Reduction of occupational fatalities and hazards.

According to the International Labour Organization [221], shipbreaking is amongst the most hazardous occupations worldwide with alarming rates of occupational fatalities, injuries, and work-related diseases. This is especially the case in less regulated shipbreaking yards where safety regulations are poorly enforced and precautions are scarcely implemented. In effect, between 2009 and 2022, there are 440 recorded worker fatalities [222] in South Asian shipbreaking yards. By introducing automated metal scrapping systems in the workflow of shipbreaking workers, there would be improved occupational safety. In particular, automated systems can reduce the worker's exposure to hazards and furthermore reduce the associated risks by distancing them from more dangerous aspects of the work. In this way, the rate of work-related fatalities, injuries, and diseases in scrapyards can be reduced and the well-being of workers can be enhanced.

8.2 Concerning Impacts

Despite the many potential benefits of introducing automation into the workflow of metal scrapping and shipbreaking yards, there are valid concerns of adverse impacts. We discuss these concerning impacts below.

Displacement of less skilled metal scrapping workers.

By integrating automated systems into a collaborative workflow with scrap cutting workers, their labor productivity would increase. As such, a smaller number of workers that supervises and collaborates with robots would be able to process a larger throughput of scrap tonnage. While it could be assumed that scrapping demand would increase sufficiently to also increase the demand for workers, there remain concerning effects. According to [223], the adoption of robots at work can impact low-skilled labor with increasing technological unemployment and depreciating wages. Conversely, for high-skilled labor, automation can lead to decreasing unemployment and appreciating wages. However, the difference between low-skill job loss and high-skill job creation is shown to vary by country. In the short term, a substantial number of less experienced and less skilled workers would be inevitably impacted with job loss or displacement. While transitioning to higher-skilled work may be possible for some, it would be infeasible for a subset of these workers—even under optimistic forecasts.

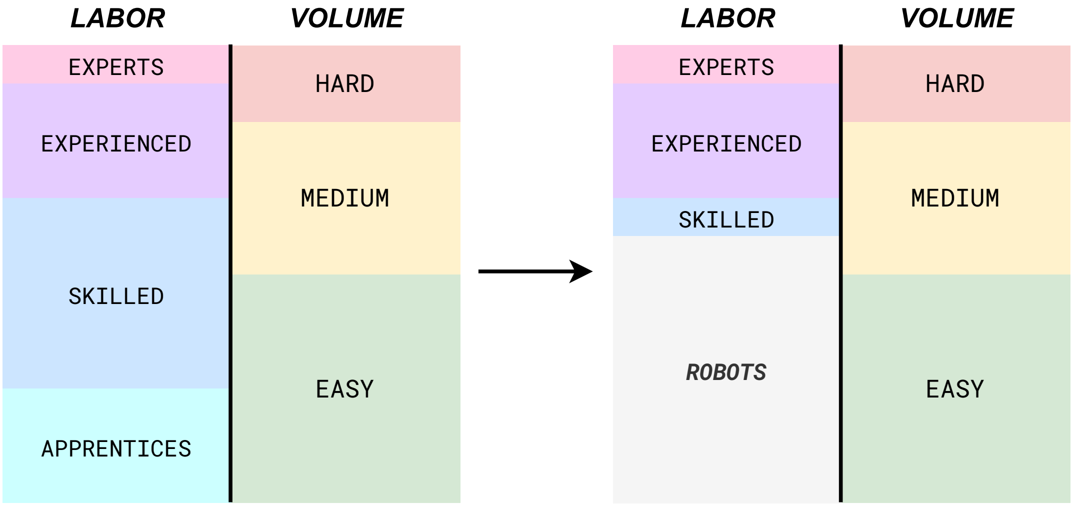


FIGURE 8.1: The adoption of robotic systems in the scrapyard environment may displace a substantial proportion of the less experienced and less skilled workers.

For illustration, we consider in Fig. 8.1 the scenario of increased adoption of automated systems in the scrapyard. The workers are ranked in decreasing order of their skill and experience, from experts to apprentices. Likewise, the volume of scrap that needs to be processed is ranked in decreasing order of difficulty. Typically, workers would process scrap pieces in accordance with their experience and skill. Experts would address cutting in the more difficult and hazardous scenarios, whereas less experienced workers would handle on the less complex cutting scenarios. In this sense, while the more experienced workers may enjoy greater productivity and safety, improved working conditions, and appreciating wages, some of the less experienced workers would endure technological unemployment.

Disrupted transfer of implicit knowledge and skill.

Metal scrap cutting is a highly kinesthetic activity where most learning and skill acquisition occurs via practice under the feedback and supervision of more experienced workers. Following from the previous concern, the displacement of the less experienced workforce in the scrapyard could result in the disrupted or reduced transfer of implicit know-how and skills. Upon the eventual retiring of the older and more experienced workforce, there would be a knowledge gap in the future generations of workers. This loss of tacit knowledge—in its most pessimistic forecast—could result in the use of automated systems whose work is poorly understood. In doing so, the dependence on poorly-understood technology could have detrimental consequences.

A related phenomenon is observed with robotic surgery and its impact [224] on traditional surgical practice and its training. More generally in [225], the disruption of expert mentorship in pursuit of greater productivity can hinder the learning of future generations, which may inevitably lead to structural problems in the future supply of expert labor.

Technological unemployment in Hispanic communities.

Within the United States, the Brownsville ship canal in Brownsville, TX is the primary scrapping hub [27] for the American shipbreaking sector. According to the United States Census Bureau [226], over 90% of Brownsville's population are of Hispanic or Latino origin as of July 2022. As such, the consequences of technological unemployment would asymmetrically impact the workforce with most of the displacement occurring in the Hispanic communities. Accordingly, we highlight the potential harm that widespread and unregulated automation in the scrapping industry may have on employment equity.

8.3 Potential Risks

In addition to the aforementioned concerning impacts, we examine further potential risks that—while particularly speculative—concern the global economy and its ecological footprint at a scale large enough to merit consideration.

Incentivization of the premature scrapping of vessels.

From the point of view of ship-owning parties, the incentive to scrap an aging vessel depends on a complex interactions of factors. Nevertheless, for the sake of discussion, we can reduce the decision-making to the following simplified dynamics:

$$\text{`Scrapping incentive'} = \text{`Projected losses'} - \text{`Scrapping cost'}$$

The implication of the above relation is that a ship-owning party is more inclined to scrap a vessel when the projected future losses of operating are higher. Furthermore, the ship-owning party is less inclined to scrap a vessel when scrapping costs are high. While this relation neither accurately nor precisely models the scrapping incentive of a ship-owning party, it nonetheless illustrates a straightforward aspect of the underlying financial management—in the interest of facilitating discussion.

With this, we can speculate on the effects of automation in shipbreaking yards. Automated metal scrapping systems would increase scrapping capacity and productivity, which may decrease overall scrapping costs. Assuming given projected losses for a particular vessel, the reduction of scrapping costs would increase the scrapping incentive of the ship-owning party. Accordingly, the vessel may be prematurely scrapped, meaning that an otherwise functional ship that may serve additional years is being recycled earlier than necessary fault of shifted financial incentives.

While this scenario is speculative in nature, it remains plausible in accordance with the observations that follow. During our on-site survey of the aforementioned shipbreaking yard, which took place in July 2021, the fallout of the COVID-19 pandemic had been affecting global shipping and maritime economics. In effect, we observed the scrapping of vessels that were described by on-site personnel as being in relatively favorable condition. One possible explanation is that the spillover of market uncertainty during COVID-19 may have increased the projected loss of operating older vessels and would have thereby increased scrapping incentives—despite a relatively unchanging scrapping cost. A related effect can be observed with the influence of government subsidies [227] on ship scrapping decisions and its tendency to increase earlier scrapping of vessels.

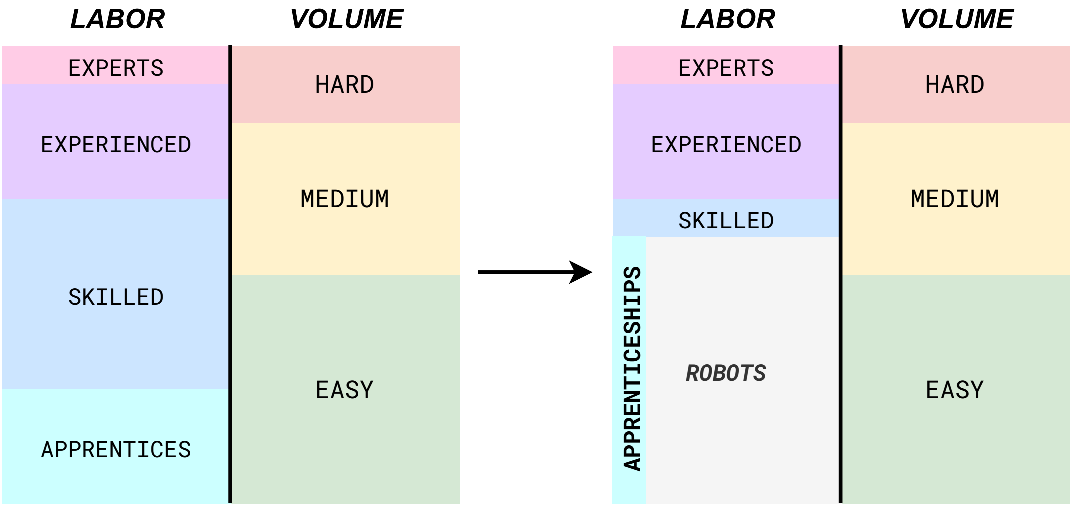


FIGURE 8.2: By sponsoring a critical number of metal scrapping apprenticeships, the tacit knowledge of the operations can be sustainably preserved across future generations of workers.

While this evidence is anecdotal in nature, it nevertheless supports the plausibility of scrapping incentives shifting as mentioned previously. In essence, excessively premature scrapping may have adverse economic and ecological impacts.

8.4 Recommendations

Following from the aforementioned concerns and risks, we provide insights and recommendations that may help in attenuating their effects.

Sponsorship of apprenticeships to preserve knowledge transfer.

To prevent or reduce the potential loss of knowledge transfer within metal scrap cutting, the industry may rely on apprenticeship, coaching, and mentoring. In effect, these learning modalities are effective vectors [228] for the transfer of tacit knowledge across generations—dating back to medieval craft guilds [229]. The importance of apprenticeship is examined in a variety of professions for transferring skills and tacit know-how such as in hairdressing [230], cuisine [231], [232], advanced academic literacy [233], and technical know-how [234].

As such, rather than unregulated automation, the industry can sponsor a critical number of apprenticeships, yielding the scenario illustrated in Fig. 8.2. These apprentices would learn and improve their tacit know-how and skills in metal scrap cutting under the guidance and supervision of more senior scrap cutting experts. By preserving a nucleus of human expertise and know-how, the industry can protect itself from loss of knowledge transfer and from the undesirable scenario of relying on machines whose work is poorly understood.

Correction of incentives using fiscal policy and taxation.

In response to the aforementioned potential risk of premature scrapping due to the automation-induced shifting of incentives, the use of fiscal policy and taxation may be used as a measure of control over such incentives. In effect, national regulators are identified [235] as the shipbreaking industry's agents of responsibility towards the implementation and enforcement of national and international recycling standards. Taxation is commonly used within this sector to influence the business dynamics and investment behavior of the scrapping industry's stakeholders.

For instance, the Chinese government has utilized tax adjustments [178]—via the issuing of tax refunds—to increase the processing of scrapped ships. Moreover, tax waivers are noted as potential incentive structures [236] for encouraging ecologically sound practices within shipbreaking.

As such, using our previously mentioned simple description of scrapping incentive, we consider the effects of a tax penalty against the premature scrapping of ships.

$$\text{`Scrapping incentive'} = \text{`Projected losses'} - (\text{`Scrapping cost'} + \text{`Tax penalty'})$$

By designing a tax penalty in accordance with the number of years a ship is being scrapped prematurely, the scrapping incentive of a ship-owning party can be dampened until the vessel has sufficiently depreciated to merit decommissioning and scrapping. In doing so, the potential economic and ecological damage of premature scrapping may be attenuated.

8.5 Further Considerations

In this section, we present more general considerations towards the ethical design of autonomous systems that are intended to operate alongside workers in complex and adverse environments—wherein safety-critical factors take precedence.

Worker participation in automation design

In developing the automation system featuring human–robot collaboration, it can be argued that involving the workers during design iterations can yield tangible benefits across the different implementation stages—from R&D to commercialization. Similar attempts are made within the framework of participatory design [237] in applications such as human–robot interaction [238] as well as in social robotics [239].

The worker can provide domain-specific expertise and tacit knowledge not readily available to researchers and system developers. But more so, the worker can routinely review the overall automation framework and may propose amendments or alternatives for greater improvements in practicality and in ergonomics. Experienced workers, in particular, can anticipate design flaws that could lead to inefficiencies, failures, or—most importantly—hazards and dangers. Such experienced workers can recall and refer to particular scenarios that they have encountered, and can extrapolate potential situations wherein the proposed solution would fail.

Furthermore, by providing workers with agency in the human–robot collaborative solution, they can empower themselves by identifying and proposing design configurations in which they can provide more value. In doing so, workers would be gradually introduced and progressively incentivized to learn robot-enabled operations, but also to innovate in such human–robot collaborative workflows, interface designs, and operational processes.

As mentioned in Section 8.2, the unchecked displacement of workers would have detrimental consequences in terms of both unemployment and knowledge transfer disruption. With the aforementioned provisions, workers would not only be protected against technological unemployment, but can also preserve the tacit knowledge of their craft across generations while progressively evolving it through inventive collaborative workflows as well as pragmatic uses of automated solutions.

Fieldwork for understanding and developing broader impacts

In Chapter 2, we highlight the importance of fieldwork as a means to better understand the problem domain, refine research objectives, and uncover crucial details for developing the relevant technological solution. Here, we emphasize the additional benefits of fieldwork pertaining to the analysis, understanding, and implementation of broader impacts. In effect, many of the key insights argued in this chapter derive from the on-site field research conducted in the aforementioned shipbreaking yard. Such fieldwork enables researchers to better understand the contextual role of the technological solution and its relationships to various stakeholders in the target domain. In doing so, the solution may be developed in ways that are more compatible with the target industry's occupational dynamics.

Additionally, a broader impacts study supported with evidence collected from the field would rely on less assumptions taken by the researcher. As such, the anticipated factors in the broader impacts study may be covered with greater breadth and depth while also retaining higher levels of confidence. In contrast, a study without sufficient evidence from the field would rely on assumptions deprived from the direct experiences of stakeholders in the targeted industry.

Holistic safety analysis for contextualizing the technological solution

In designing an automated system for safety-critical operations, it is of great importance to understand the wider context within which the system operates. In this regard, conducting a holistic safety analysis of the problem domain, its environment, and its operations enables the researchers to understand—and potentially model—the various risks and hazards therein. For instance, Chapter 2 reviews a variety of occupational hazards within the scrap cutting environment, ranging from exposure to CBRN materials to kinetic hazards (*e.g.*, falling pieces, residual explosive substances).

Within the particularities of the automation framework proposed in Chapter 3, the improvements in worker safety provided by the robotic system consist mostly in distancing the worker away from kinetic hazards as well as the heat-affected zone of oxy-fuel cutting. In contextualizing the automated system with regards to safety, its benefits as well as its limitations are more clearly understood. This allows for more conscious safety planning on a broader organizational level, where limitations can be targeted more lucidly. It is worth emphasizing that—from the point of view of the researcher—a major source of findings with regards to safety arise from observations and interactions during fieldwork.

Characterization of the technological risks

Following from the previous point on safety, the automated solution itself is not without its technological risks. In effect, the design and development of an automated system entails several assumptions which limit the extent of its use in the target domain. In deploying the system beyond its intended use, a variety of risks can arise. For instance, simplifications are often assumed during the modeling of a process to allow for tractable reasoning. Moreover, such simplified or reduced models may cover a large extent of scenarios. However, there will inevitably be scenarios in which there are model mismatches, that is, unmodeled factors such as higher-order and highly nonlinear dynamics, or external disturbances and noise.

With this in mind, the relevant stakeholders—*e.g.*, researchers, workers, and safety forepersons—must be well-informed about the nature and extent of such technological risks. To this end, quantified assessments of risk that are coarse or overly general are insufficient, while the additional understanding of failure modes becomes desirable. It is not only important to estimate the likelihood of failure, but to also capture indications of what may cause, correlate with, or result from such failures.

As such, it is beneficial to have higher levels of mechanistic insight into these failures. For example, models with white-box characteristics would be more interpretable, wherein the properties of, and relations with risks and hazards are more apparent. In contrast, models with black-box characteristics—despite often achieving higher performance—are lacking in interpretability, whereby their internal complexity may obfuscate the desired risk analysis.

With these in mind, a safety assessment can cover and characterize different types of technological risks. For instance, there are 'interpretable' risks whose failure modes are well-understood allowing for safety response protocols to be developed around them. These contrast with risks that are more poorly understood and that are less quantifiable and less predictable—similar to the distinction between risk and Knightian uncertainty [240] in economics. Additionally, there may be risks that are unanticipated or very low-probability events with severe consequences—comparable to the notion 'black swan' events, especially as applied to AI systems [241].

While interpretability and explainability are important facets in the design of intelligent agents [242], they take priority in the context of safety-critical systems. For these reasons, system developers must carefully deliberate in the tradeoff between performance and interpretability. For instance, a set of classical methods may be more apt for the most safety-critical tasks despite their potentially lower accuracy. In tandem, potentially better performance from learning-based methods should be not be used at the expense of safety considerations.

8.6 Conclusion

With this anticipatory analysis, we have highlighted the potential benefits, concerns, and risks of adopting automated scrapping systems. Furthermore, we argued for recommendations to partially address some of these concerns and risks. Accordingly, technological progress within automated metal scrap cutting could be received—and potentially informed—with better preparedness and direction towards more sustainable socio-economic outcomes. Finally, we presented further considerations for broader impacts that may be extendable to the ethical design of other autonomous systems; especially ones to be operated in adverse environments alongside workers.

Chapter 9

Conclusion

In this dissertation, we designed a framework and its underlying methods for the purpose of endowing robotic systems with autonomous functionalities tailored for the scrap cutting tasks of the metal scrapyard. To this end, we introduced the metal scrap recycling industry and its problems as well as the opportunities and challenges pertaining to its automation. Additionally, we provided many critical insights through the on-site surveying of a representative shipbreaking yard and its operations. From these, we formulated our research objectives and formalized the specific research problems of our proposed framework. These problems are: 'Cutting path generation', 'Autonomous oxy-fuel cutting', 'Task state monitoring', and 'Safe structural disassembly'.

In our framework, we designed a human–robot collaboration workflow and argued its advantages in leveraging the unique strengths of both the worker and the robot—whose configuration is described. We furthermore presented our framework's architecture, its components, and the interaction between them. Accordingly, we supplied a review of these components' underlying methods and techniques, namely: Viewpoint planning, vision-based control, neural network-based classification, and sequential decision-making. Through these, we designed, developed, and evaluated our methodologies for the aforementioned specific research problems. Afterwards, we performed an anticipatory ethical analysis highlighting the broader impacts of this research in terms of benefits, concerns, and potential risks—as well as suggesting recommendations for attenuating some of these concerns and risks.

Ultimately, through our treatment of developing robot autonomy targeting metal recycling and its operations, we demonstrated the motive and advantage of using a diverse cognitive architecture for taming the difficulty within adverse environments.

9.1 Summary of the Proposed Methods

For cutting path generation (Chapter 4), we proposed feature-driven next view planning—alongside human–robot collaboration—as an effective active vision scheme for generating cutting paths on arbitrary object surfaces. To this end, we devised three next view planning algorithms and evaluated their performance both in simulation and in physical experiments against six categories of object surfaces. In particular, our third algorithm (G-NBV) achieved successful, efficient, and robust feature reconstruction against all test objects.

For autonomous oxy-fuel cutting (Chapter 5), we proposed a formal treatment of the cutting problem for the purpose of its automation. We designed a strategy for vision-based torch control inspired by the cutting techniques of skilled cutting workers in the surveyed scrapyard. The visual feedback consists of processing the eye-in-hand view of the heated region on the surface to extract meaningful features for cutting. These are heat pool convexity and intensity which distill the pool's shape, size, color, and brightness. These features are used to express the pool combustion state, which is tracked during control to reach its desired value by appropriately regulating the torch's motion. We evaluated our control scheme in physical experiments achieving the successful cutting of steel plates having different thicknesses.

For task state monitoring (Chapter 6), we motivated the need to estimate the task state of an automated cutting system—for the purpose of monitoring its process and improving its safety. To this end, we curated our open-access 'Oxy-fuel Cutting Task State Image Dataset' using footage from 50 recorded cutting experiments. With this dataset, we designed and implemented a convolutional neural network model for classifying input images into one of four task states. We evaluated this classifier against test data, achieving an overall accuracy of 93.8%. In addition, we achieved fast inference speeds on two representative computing hardware, thus demonstrating the classifier's feasibility for online monitoring.

Finally, for safe structural disassembly (Chapter 7), we presented the problem formulation on partitioned input structures. For this, we designed a mathematical model for safety that considers both the dynamic and the geometric effects of moving segments. Accordingly, we devised our decision algorithm that makes use of physics-based simulations to assess the outcomes of a decision and thereafter measure its safety. We developed a greedy decision search scheme to overcome the growing number of decision evaluations (relative to input size) while maintaining acceptable safety scores. The performance of our greedy scheme is compared against exhaustive and random schemes against 12 simulated structures. Our evaluations demonstrate that our greedy scheme obtains decision sequences efficiently and with high safety scores.

9.2 Recommendations for Future Research

In light of the insights gained through conducting this research, we share the following recommendations. For robotics research targeting applied and domain-specific settings—especially environmental robotics—it is critical to analyze the target domain and gain an accurate understanding of its unique needs and challenges prior to the up-scaling of framework designs and methodologies. Researchers and practitioners should emphasize 'problem definition' [243] and integrate it into 'problem solving'.

To this end, we recommend the surveying of the target domain's operations through on-site fieldwork. This would enable the deconstruction of a larger entangled problem towards the identification of specific research problems. Furthermore, by understanding the specific needs of each task, the researcher can more easily assess the adequacy of chosen methodologies. In doing so, the research objectives would become more relevant and effective towards achieving the wider needs of the target domain.

More generally, while robotics is increasingly diverse [244], we further advocate for the interdisciplinarity [245] of its research. Through collaboration, robotics research—with its expanding technological convergence [246]—would more inclusively reflect society's possibilities and values [247] towards brighter futures.

Bibliography

[1] G. De Luca, “The development of machine intelligence in a computational universe,” *Technology in Society*, vol. 65, p. 101 553, 2021, ISSN: 0160-791X. DOI: 10.1016/j.techsoc.2021.101553.

[2] P. McCorduck and C. Cfe, *Machines Who Think: A Personal Inquiry into the History and Prospects of Artificial Intelligence*. A K Peters / CRC Press, 2004, ISBN: 1568812051. DOI: 10.1201/9780429258985.

[3] E. Garcia, M. A. Jimenez, P. G. De Santos, and M. Armada, “The evolution of robotics research,” *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 90–103, 2007. DOI: 10.1109/MRA.2007.339608.

[4] G. O’Regan, “Unimation,” in *Pillars of Computing: A Compendium of Select, Pivotal Technology Firms*, Cham: Springer International Publishing, 2015, pp. 219–223. DOI: 10.1007/978-3-319-21464-1_34.

[5] B. Kuipers, E. A. Feigenbaum, P. E. Hart, and N. J. Nilsson, “Shakey: From conception to history,” *AI Magazine*, vol. 38, no. 1, pp. 88–103, Mar. 2017. DOI: 10.1609/aimag.v38i1.2716.

[6] C. J. C. H. Watkins, “Learning from delayed rewards,” 1989. [Online]. Available: https://www.academia.edu/download/50360235/Learning_from_delayed_rewards_20161116-28282-v2pwvq.pdf.

[7] J. Matijevic, “Sojourner: The mars pathfinder microrover flight experiment,” English, *Space Technology*, vol. 17, no. 3-4, pp. 143–149, 1997.

[8] S. M. LaValle, “Rapidly-exploring random trees : A new tool for path planning,” *The annual research report*, 1998. [Online]. Available: http://msl.cs.illinois.edu/~lavalle/papers/Lav98c.pdf.

[9] J. Jones, “Robots at the tipping point: The road to iRobot Roomba,” *IEEE Robotics & Automation Magazine*, vol. 13, no. 1, pp. 76–78, 2006. DOI: 10.1109/MRA.2006.1598056.

[10] S. Thrun, M. Montemerlo, H. Dahlkamp, *et al.*, “Stanley: The robot that won the DARPA Grand Challenge,” *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006. DOI: 10.1002/rob.20147.

[11] E. Guizzo and E. Ackerman, “The rise of the robot worker,” *IEEE Spectrum*, vol. 49, no. 10, pp. 34–41, 2012. DOI: 10.1109/MSPEC.2012.6309254.

[12] P. K. LeHardy and C. Moore, “Deep ocean search for Malaysia airlines flight 370,” in *2014 Oceans – St. John’s*, 2014, pp. 1–4. DOI: 10.1109/OCEANS.2014.7003292.

[13] J. M. Watts, “World’s first self-driving taxis hit the road in Singapore,” *The Wall Street Journal*, 2016. [Online]. Available: https://www.belegger.nl/Forum/Upload/2016/9547734.pdf.

[14] J. H. Schut, “The challenge of recycling ’problem’ packages,” *Plastics Engineering*, vol. 73, no. 6, pp. 26–29, 2017. DOI: 10.1002/j.1941-9635.2017.tb01727.x.

[15] M. Cardona, A. Palma, and J. Manzanares, "COVID-19 pandemic impact on mobile robotics market," in *2020 IEEE ANDESCON*, 2020, pp. 1–4. DOI: 10.1109/ANDESCON50619.2020.9272052.

[16] A. Witze, "Lift off! First flight on Mars launches new way to explore worlds," *Nature*, vol. 592, no. 7856, pp. 668–669, 2021. DOI: 10.1038/d41586-021-00909-z.

[17] C. Wong, E. Yang, X.-T. Yan, and D. Gu, "Autonomous robots for harsh environments: A holistic overview of current solutions and ongoing challenges," *Systems Science & Control Engineering*, vol. 6, no. 1, pp. 213–219, Jan. 2018. DOI: 10.1080/21642583.2018.1477634.

[18] P. S. Rosenbloom, "Rethinking cognitive architecture via graphical models," *Cognitive Systems Research*, vol. 12, no. 2, pp. 198–209, 2011, ISSN: 1389-0417. DOI: 10.1016/j.cogsys.2010.07.006.

[19] P. Söderholm and T. Ekvall, "Metal markets and recycling policies: Impacts and challenges," *Mineral Economics*, vol. 33, no. 1, pp. 257–272, 2020, ISSN: 2191-2211. DOI: 10.1007/s13563-019-00184-5.

[20] L. D. Harvey, "Iron and steel recycling: Review, conceptual model, irreducible mining requirements, and energy implications," *Renewable and Sustainable Energy Reviews*, vol. 138, p. 110 553, 2021, ISSN: 1364-0321. DOI: 10.1016/j.rser.2020.110553.

[21] K. P. Jain and J. Pruyn, "An overview of the global ship recycling industry," in *Encyclopedia of Renewable and Sustainable Materials*, S. Hashmi and I. A. Choudhury, Eds., Oxford: Elsevier, 2017, pp. 626–647, ISBN: 978-0-12-813196-1. DOI: 10.1016/B978-0-12-803581-8.10396-0.

[22] J.-P. Rodrigue, "The Vulnerability and Resilience of the Global Container Shipping Industry," *Current History*, vol. 121, no. 831, pp. 17–23, Jan. 2022, ISSN: 0011-3530. DOI: 10.1525/curh.2022.121.831.17.

[23] X. Gou and J. S. L. Lam, "Risk analysis of marine cargoes and major port disruptions," *Maritime Economics & Logistics*, vol. 21, no. 4, pp. 497–523, 2019, ISSN: 1479-294X. DOI: 10.1057/s41278-018-0110-3.

[24] M. Menhat, I. M. Mohd Zaideen, Y. Yusuf, N. H. M. Salleh, M. A. Zamri, and J. Jeevan, "The impact of Covid-19 pandemic: A review on maritime sectors in malaysia," *Ocean & Coastal Management*, vol. 209, p. 105 638, 2021, ISSN: 0964-5691. DOI: 10.1016/j.ocecoaman.2021.105638.

[25] J. Yin and L. Fan, "Survival analysis of the world ship demolition market," *Transport Policy*, vol. 63, pp. 141–156, 2018, ISSN: 0967-070X. DOI: 10.1016/j.tranpol.2017.12.019.

[26] M. J. Kaiser, "A review of ship breaking and rig scrapping in the gulf of mexico," *Ocean Development & International Law*, vol. 39, no. 2, pp. 178–199, May 2008, ISSN: 0090-8320. DOI: 10.1080/00908320802013701.

[27] M. Galley, "Industry development and the process of disposal," in *Shipbreaking: Hazards and Liabilities*, Cham: Springer International Publishing, 2014, pp. 1–42. DOI: 10.1007/978-3-319-04699-0_1.

[28] S. Knapp, S. N. Kumar, and A. B. Remijn, "Econometric analysis of the ship demolition market," *Marine Policy*, vol. 32, no. 6, pp. 1023–1036, 2008, ISSN: 0308-597X. DOI: 10.1016/j.marpol.2008.02.004.

[29] O. Dinu and A. M. Ilie, "Maritime vessel obsolescence, life cycle cost and design service life," *IOP Conference Series: Materials Science and Engineering*, vol. 95, no. 1, p. 012 067, Oct. 2015. DOI: 10.1088/1757-899X/95/1/012067.

[30] Z. Du, S. Zhang, Q. Zhou, K. F. Yuen, and Y. D. Wong, "Hazardous materials analysis and disposal procedures during ship recycling," *Resources,*

Conservation and Recycling, vol. 131, pp. 158–171, 2018, ISSN: 0921-3449. DOI: 10.1016/j.resconrec.2018.01.006.

[31] A. Rzeźnikiewicz, "Cost comparison between oxyfuel and plasma cutting low alloy steel," *Journal of Achievements in Materials and Manufacturing Engineering*, vol. Vol. 63, nr 2, pp. 81–85, 2014. [Online]. Available: http://jamme.acmsse.h2.pl/vol63_2/6324.pdf.

[32] P. Rousmaniere and N. Raj, "Shipbreaking in the developing world: Problems and prospects," *International Journal of Occupational and Environmental Health*, vol. 13, no. 4, pp. 359–368, Oct. 2007, ISSN: 1077-3525. DOI: 10.1179/oeh.2007.13.4.359.

[33] NGO Shipbreaking Platform, *Impact report 2020 – 2021*, Nov. 2022. [Online]. Available: https://shipbreakingplatform.org/wp-content/uploads/2022/11/NGO-SBP-Annual-Report-2020_2021.pdf.

[34] Occupational Safety and Health Administration (OSHA), *OSHA 3348-05 Guidance for the identification and control of safety and health hazards in metal scrap recycling*, 2008. [Online]. Available: https://www.osha.gov/sites/default/files/publications/OSHA3348-metal-scrap-recycling.pdf.

[35] The National Institute for Occupational Safety and Health (NIOSH), *Hispanic scrap yard worker dies when struck by material handler at metal recycling facility—South Carolina*, Dec. 2015. [Online]. Available: https://www.cdc.gov/niosh/face/in-house/full201301.html.

[36] New York State Department of Health, *Metal recycling worker run over by a front-end loader at a scrap yard (Case #17NY023)*, Jun. 2017. [Online]. Available: https://www.health.ny.gov/environmental/investigations/face/docs/17ny023.pdf.

[37] A. M. Hiremath, A. K. Tilwankar, and S. R. Asolekar, "Significant steps in ship recycling vis-a-vis wastes generated in a cluster of yards in Alang: A case study," *Journal of Cleaner Production*, vol. 87, pp. 520–532, 2015, ISSN: 0959-6526. DOI: 10.1016/j.jclepro.2014.09.031.

[38] A. M. Hiremath, S. K. Pandey, and S. R. Asolekar, "Development of ship-specific recycling plan to improve health safety and environment in ship recycling yards," *Journal of Cleaner Production*, vol. 116, pp. 279–298, 2016, ISSN: 0959-6526. DOI: https://doi.org/10.1016/j.jclepro.2016.01.006.

[39] U.S. Environmental Protection Agency (EPA), *A guide for ship scrappers: Tips for regulatory compliance (EPA 315-B-00-001)*, 2000. [Online]. Available: https://nepis.epa.gov/Exe/ZyPDF.cgi/50000M6P.PDF?Dockey=50000M6P.PDF.

[40] T. Karlis and D. Polemis, "Ship demolition activity: A monetary flow process approach," *Pomorstvo*, vol. 30, no. 2, pp. 128–132, 2016. DOI: 10.31217/p.30.2.5.

[41] T. Karlis, D. Polemis, and A. Georgakis, "Ship demolition activity. an evaluation of the effect of currency exchange rates on ship scrap values.," *SPOUDAI – Journal of Economics and Business*, vol. 66, no. 3, pp. 53–70, 2016, ISSN: 2241-424X. [Online]. Available: https://spoudai.unipi.gr/index.php/spoudai/article/view/2551.

[42] X. Kong, K. Feng, P. Wang, *et al.*, "Steel stocks and flows of global merchant fleets as material base of international trade from 1980 to 2050," *Global Environmental Change*, vol. 73, p. 102 493, 2022, ISSN: 0959-3780. DOI: 10.1016/j.gloenvcha.2022.102493.

[43] S. M. Rahman, J. Kim, and B. Laratte, "Disruption in circularity? impact analysis of COVID-19 on ship recycling using Weibull tonnage estimation

and scenario analysis method," *Resources, Conservation and Recycling*, vol. 164, p. 105 139, 2021, ISSN: 0921-3449. DOI: 10.1016/j.resconrec.2020.105139.

[44] W. Shen and G. Xing, "Study on status quo of shipbreaking sector and strategies," *Iop Conference Series: Earth and Environmental Science*, vol. 94, no. 1, p. 012 163, Nov. 2017. DOI: 10.1088/1755-1315/94/1/012163.

[45] T. Watari, K. Nansai, and K. Nakajima, "Major metals demand, supply, and environmental impacts to 2100: A critical review," *Resources, Conservation and Recycling*, vol. 164, p. 105 107, 2021, ISSN: 0921-3449. DOI: 10.1016/j.resconrec.2020.105107.

[46] M. N. O. Sadiku, A. J. Ajayi-Majebi, P. O. Adebo, M. N. O. Sadiku, A. J. Ajayi-Majebi, and P. O. Adebo, "Robotic automation in manufacturing," in *Emerging Technologies in Manufacturing*, Cham: Springer International Publishing, 2023, pp. 33–47. DOI: 10.1007/978-3-031-23156-8_3.

[47] G. Q. Huang, M. Z. Q. Chen, and J. Pan, "Robotics in ecommerce logistics," *HKIE Transactions*, vol. 22, no. 2, pp. 68–77, Apr. 2015, ISSN: 1023-697X. DOI: 10.1080/1023697X.2015.1043960.

[48] S. D. Han, B. Huang, S. Ding, *et al.*, "Toward fully automated metal recycling using computer vision and non-prehensile manipulation," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, 2021, pp. 891–898. DOI: 10.1109/CASE49439.2021.9551431.

[49] R. Bogue, "Cutting robots: A review of technologies and applications," *Industrial Robot: An International Journal*, vol. 35, no. 5, J. Franks, Ed., pp. 390–396, Jan. 2008, ISSN: 0143-991X. DOI: 10.1108/01439910810893554.

[50] S. Gasper and J. Bickendorf, "Robot cutting in ship building industry – a new flexible approach for linking parametric design and fully automatic robot programming," in *ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*, 2010, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/document/5756856/.

[51] J. Bickendorf, "New applications of cutting and welding robots with automatic offline-programming," in *Proceedings of ISR 2016: 47st International Symposium on Robotics*, 2016, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7559179.

[52] B. Denkena and T. Lepper, "Enabling an industrial robot for metal cutting operations," *Procedia CIRP*, vol. 35, pp. 79–84, 2015, MIC2015 – 15th Machining Innovations Conference for Aerospace Industry, ISSN: 2212-8271. DOI: 10.1016/j.procir.2015.08.100.

[53] *Hybrid induction plasma-oxygen cutting, using a mobile robot system for shipyard application*, vol. Day 3 Thu, November 03, 2016, SNAME Maritime Convention, D033S010R006, Nov. 2016. [Online]. Available: https://onepetro.org/SNAMESMC/proceedings-abstract/SMC16/3-SMC16/D033S010R006/21246.

[54] K. Yoo, H. Ryu, and C. Choi, "Control architecture design for an gas cutting robot," in *2008 Second International Conference on Future Generation Communication and Networking Symposia*, vol. 4, 2008, pp. 66–71. DOI: 10.1109/FGCNS.2008.130.

[55] J. Akl, F. Alladkani, and B. Calli, "Feature-driven next view planning for cutting path generation in robotic metal scrap recycling," *IEEE Transactions on Automation Science and Engineering*, pp. 1–17, 2023. DOI: 10.1109/TASE.2023.3278994.

[56] J. Akl, F. Alladkani, and B. Calli, "Towards robotic metal scrap cutting: A novel workflow and pipeline for cutting path generation," in *2021 IEEE 17th*

International Conference on Automation Science and Engineering (CASE), 2021, pp. 132–137. DOI: 10.1109/CASE49439.2021.9551645.

[57] V. Villani, F. Pini, F. Leali, and C. Secchi, "Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications," *Mechatronics*, vol. 55, pp. 248–266, 2018, ISSN: 0957-4158. DOI: 10.1016/j.mechatronics.2018.02.009.

[58] R. Zeng, Y. Wen, W. Zhao, and Y.-J. Liu, "View planning in robot active vision: A survey of systems, algorithms, and applications," *Computational Visual Media*, vol. 6, no. 3, pp. 225–245, 2020, ISSN: 2096-0662. DOI: 10.1007/s41095-020-0179-3.

[59] S. Chen, Y. F. Li, J. Zhang, and W. Wang, "Information entropy based planning," in *Active Sensor Planning for Multiview Vision Tasks*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 147–176. DOI: 10.1007/978-3-540-77072-5_8.

[60] P. Corke, "Vision-based control," in *Robotics, Vision and Control*, Cham: Springer International Publishing, 2017, pp. 537–563. DOI: 10.1007/978-3-319-54413-7_15.

[61] W. E, C. Ma, L. Wu, and S. Wojtowytsch, "Towards a mathematical understanding of neural network-based machine learning: What we know and what we don't," *CSIAM Transactions on Applied Mathematics*, vol. 1, no. 4, pp. 561–615, 2020, ISSN: 2708-0579. DOI: 10.4208/csiam-am.SO-2020-0002.

[62] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Computation*, vol. 29, no. 9, pp. 2352–2449, 2017. DOI: 10.1162/neco_a_00990.

[63] M. J. Kochenderfer, "Sequential problems," in *Decision Making Under Uncertainty: Theory and Application*, The MIT Press, Jul. 2015. DOI: 10.7551/mitpress/10187.003.0008.

[64] J. Daudelin and M. Campbell, "An adaptable, probabilistic, next-best view algorithm for reconstruction of unknown 3-D objects," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1540–1547, 2017. DOI: 10.1109/LRA.2017.2660769.

[65] S. Isler, R. Sabzevari, J. Delmerico, and D. Scaramuzza, "An information gain formulation for active volumetric 3D reconstruction," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 3477–3484. DOI: 10.1109/ICRA.2016.7487527.

[66] R. Dewil, P. Vansteenwegen, and D. Cattrysse, "A review of cutting path algorithms for laser cutters," *The International Journal of Advanced Manufacturing Technology*, vol. 87, no. 5, pp. 1865–1884, 2016, ISSN: 1433-3015. DOI: 10.1007/s00170-016-8609-1.

[67] H. Mei, Z. Li, and C. Zou, "Automatic cutting system design of robot hand based on stereo vision," *Scientific Programming*, vol. 2022, p. 4 663 213, 2022, ISSN: 1058-9244. DOI: 10.1155/2022/4663213.

[68] Z. Xu, M. Liang, X. Fang, G. Wu, N. Chen, and Y. Song, "Research on autonomous cutting method of cantilever roadheader," *Energies*, vol. 15, no. 17, 2022, ISSN: 1996-1073. DOI: 10.3390/en15176190.

[69] D. Lindberget, *Automatic generation of robot targets: A first step towards a flexible robotic solution for cutting customized mesh tray*, 2022. [Online]. Available: http://hv.diva-portal.org/smash/record.jsf?pid=diva2%3A1690760.

[70] J. Cai, Z. Ding, Y. Zhang, and M. Liu, "Trajectory planning and simulation for intersecting line cutting of the industry robot," in *Proceeding of the 11th*

World Congress on Intelligent Control and Automation, 2014, pp. 63–68. DOI: 10.1109/WCICA.2014.7052688.

[71] Z. Xia, K. Zhou, X. Li, and X. Cui, "A method of robot laser cutting for small holes," in *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, 2016, pp. 1887–1891. DOI: 10.1109/CISP-BMEI.2016.7853025.

[72] J. Hatwig, P. Minnerup, M. F. Zaeh, and G. Reinhart, "An automated path planning system for a robot with a laser scanner for remote laser cutting and welding," in *2012 IEEE International Conference on Mechatronics and Automation*, 2012, pp. 1323–1328. DOI: 10.1109/ICMA.2012.6284328.

[73] P. Yang, T. Lei, C. Wu, S. Zhao, and J. Hu, "A fast calibration of laser vision robotic welding systems using automatic path planning," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–10, 2022. DOI: 10.1109/TIM.2022.3214632.

[74] Z. Hou, Y. Xu, R. Xiao, and S. Chen, "A teaching-free welding method based on laser visual sensing system in robotic GMAW," *The International Journal of Advanced Manufacturing Technology*, vol. 109, no. 5, pp. 1755–1774, 2020, ISSN: 1433-3015. DOI: 10.1007/s00170-020-05774-0.

[75] L. Yang, E. Li, T. Long, J. Fan, and Z. Liang, "A novel 3-D path extraction method for arc welding robot based on stereo structured light sensor," *IEEE Sensors Journal*, vol. 19, no. 2, pp. 763–773, 2019. DOI: 10.1109/JSEN.2018.2877976.

[76] J. Muhammad, H. Altun, and E. Abo-Serie, "Welding seam profiling techniques based on active vision sensing for intelligent robotic welding," *The International Journal of Advanced Manufacturing Technology*, vol. 88, no. 1, pp. 127–145, 2017, ISSN: 1433-3015. DOI: 10.1007/s00170-016-8707-0.

[77] Y. Xu, G. Fang, S. Chen, J. J. Zou, and Z. Ye, "Real-time image processing for vision-based weld seam tracking in robotic GMAW," *The International Journal of Advanced Manufacturing Technology*, vol. 73, no. 9, pp. 1413–1425, 2014, ISSN: 1433-3015. DOI: 10.1007/s00170-014-5925-1.

[78] M. Dinham and G. Fang, "Autonomous weld seam identification and localisation using eye-in-hand stereo vision for robotic arc welding," *Robotics and Computer-Integrated Manufacturing*, vol. 29, no. 5, pp. 288–301, 2013, ISSN: 0736-5845. DOI: 10.1016/j.rcim.2013.01.004.

[79] Z. Ye, G. Fang, S. Chen, and M. Dinham, "A robust algorithm for weld seam extraction based on prior knowledge of weld seam," *Sensor Review*, vol. 33, no. 2, pp. 125–133, Jan. 2013, ISSN: 0260-2288. DOI: 10.1108/02602281311299662.

[80] Q. Wang, Y. Cheng, W. Jiao, M. T. Johnson, and Y. Zhang, "Virtual reality human-robot collaborative welding: A case study of weaving gas tungsten arc welding," *Journal of Manufacturing Processes*, vol. 48, pp. 210–217, 2019, ISSN: 1526-6125. DOI: 10.1016/j.jmapro.2019.10.016.

[81] L. Hou, X. Chen, K. Lan, R. Rasmussen, and J. Roberts, "Volumetric next best view by 3D occupancy mapping using Markov chain Gibbs sampler for precise manufacturing," *IEEE Access*, vol. 7, pp. 121 949–121 960, 2019. DOI: 10.1109/ACCESS.2019.2935547.

[82] W. Peng, Y. Wang, Z. Miao, M. Feng, and Y. Tang, "Viewpoints planning for active 3-D reconstruction of profiled blades using estimated occupancy probabilities (EOP)," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 5, pp. 4109–4119, 2021. DOI: 10.1109/TIE.2020.2987286.

[83] S. Kiciroglu, H. Rhodin, S. N. Sinha, M. Salzmann, and P. Fua, "ActiveMoCap: Optimized viewpoint selection for active human motion capture," in *2020*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 100–109. DOI: 10.1109/CVPR42600.2020.00018.

[84] N. Palomeras, N. Hurtós, E. Vidal, and M. Carreras, "Autonomous exploration of complex underwater environments using a probabilistic next-best-view planner," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1619–1625, 2019. DOI: 10.1109/LRA.2019.2896759.

[85] S. Natarajan, G. Brown, and B. Calli, "Aiding grasp synthesis for novel objects using heuristic-based and data-driven active vision methods," *Frontiers in Robotics and AI*, vol. 8, 2021, ISSN: 2296-9144. DOI: 10.3389/frobt.2021.696587.

[86] M. Hegedus, K. Gupta, and M. Mehrandezh, "Towards an integrated autonomous data-driven grasping system with a mobile manipulator," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 1601–1607. DOI: 10.1109/ICRA.2019.8793759.

[87] D. Stogl, D. Zumkeller, S. E. Navarro, A. Heilig, and B. Hein, "Tracking, reconstruction and grasping of unknown rotationally symmetrical objects from a conveyor belt," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2017, pp. 1–8. DOI: 10.1109/ETFA.2017.8247651.

[88] S. Obwald and M. Bennewitz, "GPU-accelerated next-best-view coverage of articulated scenes," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 603–610. DOI: 10.1109/IROS.2018.8594054.

[89] L. Xu, W. Cheng, K. Guo, L. Han, Y. Liu, and L. Fang, "FlyFusion: Realtime dynamic scene reconstruction using a flying depth camera," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 1, pp. 68–82, 2021. DOI: 10.1109/TVCG.2019.2930691.

[90] R. Border and J. D. Gammell, "Proactive estimation of occlusions and scene coverage for planning next best views in an unstructured representation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 4219–4226. DOI: 10.1109/IROS45743.2020.9341681.

[91] R. Zeng, W. Zhao, and Y.-J. Liu, "PC-NBV: A point cloud based deep network for efficient next best view planning," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 7050–7057. DOI: 10.1109/IROS45743.2020.9340916.

[92] C. Wu, R. Zeng, J. Pan, C. C. L. Wang, and Y.-J. Liu, "Plant phenotyping by deep-learning-based planner for multi-robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3113–3120, 2019. DOI: 10.1109/LRA.2019.2924125.

[93] R. Monica and J. Aleotti, "A probabilistic next best view planner for depth cameras based on deep learning," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3529–3536, 2021. DOI: 10.1109/LRA.2021.3064298.

[94] C. Collander, W. J. Beksi, and M. Huber, "Learning the next best view for 3D point clouds via topological features," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 12 207–12 213. DOI: 10.1109/ICRA48506.2021.9561389.

[95] K. Armijo, T. McCarthy, E. Jackson, and A. Corey, "Design and implementation of a modular mobile robotic base," Tech. Rep., 2022. [Online]. Available: https://digital.wpi.edu/show/h702q9959.

[96] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 1–4. DOI: 10.1109/ICRA.2011.5980567.

[97] M. Quigley, K. Conley, B. Gerkey, *et al.*, "ROS: An open-source robot operating system," in *ICRA Workshop on Open Source Software*, vol. 3, Jan. 2009. [Online]. Available: http://robotics.stanford.edu/~ang/papers/icraoss09-ROS.pdf.

[98] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No. 04CH37566)*, vol. 3, 2004, pp. 2149–2154. DOI: 10.1109/IROS.2004.1389727.

[99] D. Coleman, I. A. Șucan, S. Chitta, and N. Correll, "Reducing the barrier to entry of complex robotic software: A MoveIt! case study," *Journal of Software Engineering for Robotics*, vol. 5, no. 1, pp. 3–16, May 2014, ISSN: 0921-3449. DOI: 10.6092/JOSER_2014_05_01_p3.

[100] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013, ISSN: 1573-7527. DOI: 10.1007/s10514-012-9321-0.

[101] W. Zheng, P. Bo, Y. Liu, and W. Wang, "Fast B-spline curve fitting by L-BFGS," *Computer Aided Geometric Design*, vol. 29, no. 7, pp. 448–462, 2012, Geometric Modeling and Processing 2012, ISSN: 0167-8396. DOI: 10.1016/j.cagd.2012.03.004.

[102] W. Wang, H. Pottmann, and Y. Liu, "Fitting B-spline curves to point clouds by curvature-based squared distance minimization," *ACM Transactions on Graphics*, vol. 25, no. 2, pp. 214–238, Apr. 2006, ISSN: 0730-0301. DOI: 10.1145/1138450.1138453.

[103] L. Fang and D. C. Gossard, "Multidimensional curve fitting to unorganized data points by nonlinear minimization," *Computer-Aided Design*, vol. 27, no. 1, pp. 48–58, 1995, ISSN: 0010-4485. DOI: 10.1016/0010-4485(95)90752-2.

[104] S. Flöry, "Fitting curves and surfaces to point clouds in the presence of obstacles," *Computer Aided Geometric Design*, vol. 26, no. 2, pp. 192–202, 2009, ISSN: 0167-8396. DOI: 10.1016/j.cagd.2008.04.003.

[105] K. Khanna and N. Rajpal, "Reconstruction of curves from point clouds using fuzzy logic and ant colony optimization," *Neurocomputing*, vol. 161, pp. 72–80, 2015, ISSN: 0925-2312. DOI: 10.1016/j.neucom.2014.11.071.

[106] T. Hastie and W. Stuetzle, "Principal curves," *Journal of the American Statistical Association*, vol. 84, no. 406, pp. 502–516, 1989. DOI: 10.1080/01621459.1989.10478797.

[107] B. Kegl and A. Krzyzak, "Piecewise linear skeletonization using principal curves," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 59–74, 2002. DOI: 10.1109/34.982884.

[108] T. Mörwald, *Fitting a 3D B-spline curve to point-clouds using point-distance-minimization and optionally asymmetric-distance-minimization*. [Online]. Available: https://github.com/PointCloudLibrary/pcl/blob/master/surface/include/pcl/surface/on_nurbs/fitting_curve_pdm.h.

[109] D. Jin, K. S. Iyer, C. Chen, E. A. Hoffman, and P. K. Saha, "A robust and efficient curve skeletonization algorithm for tree-like objects using minimum cost paths," *Pattern Recognition Letters*, vol. 76, pp. 32–40, 2016, Special Issue on Skeletonization and its Application, ISSN: 0167-8655. DOI: 10.1016/j.patrec.2015.04.002.

[110] K. Saeed, M. Tabędzki, M. Rybnik, and M. Adamski, "K3M: A universal algorithm for image skeletonization and a review of thinning techniques,"

International Journal of Applied Mathematics and Computer Science, vol. 20, no. 2, pp. 317–335, 2010. DOI: 10.2478/v10006-010-0024-4.

[111] S. Tran and L. Shih, "Efficient 3D binary image skeletonization," in *2005 IEEE Computational Systems Bioinformatics Conference – Workshops (CSBW'05)*, 2005, pp. 364–372. DOI: 10.1109/CSBW.2005.57.

[112] T. Lee, R. Kashyap, and C. Chu, "Building skeleton models via 3-D medial surface axis thinning algorithms," *CVGIP: Graphical Models and Image Processing*, vol. 56, no. 6, pp. 462–478, 1994, ISSN: 1049-9652. DOI: 10.1006/cgip.1994.1042.

[113] Intel Corporation, *Intel RealSense D400 series product family datasheet*, Jan. 2019. [Online]. Available: https://www.intel.com/content/dam/support/us/en/documents/emerging-technologies/intel-realsense-technology/Intel-RealSense-D400-Series-Datasheet.pdf.

[114] J. Akl, Y. Patil, C. Todankar, and B. Calli, *Vision-based oxy-fuel torch control for robotic metal cutting*, 2023. arXiv: 2307.00133 [cs.RO].

[115] G. Graetz and G. Michaels, "Robots at work," *The Review of Economics and Statistics*, vol. 100, no. 5, pp. 753–768, Dec. 2018, ISSN: 0034-6535. DOI: 10.1162/rest_a_00754.

[116] R. Gihleb, O. Giuntella, L. Stella, and T. Wang, "Industrial robots, workers' safety, and health," *Labour Economics*, vol. 78, p. 102 205, 2022, ISSN: 0927-5371. DOI: 10.1016/j.labeco.2022.102205.

[117] Y. Chen, L. Cheng, and C.-C. Lee, "How does the use of industrial robots affect the ecological footprint? international evidence," *Ecological Economics*, vol. 198, p. 107 483, 2022, ISSN: 0921-8009. DOI: 10.1016/j.ecolecon.2022.107483.

[118] K. Nachbargauer, "Oxy-fuel cutting: Automation makes the difference," *Zavarivanje i zavarene konstrukcije*, vol. 64, no. 1, pp. 39–45, 2019. DOI: 10.5937/zzk1901039N.

[119] I. Iglesias, M. A. Sebastián, and J. E. Ares, "Overview of the state of robotic machining: Current situation and future potential," *Procedia Engineering*, vol. 132, pp. 911–917, 2015, MESIC Manufacturing Engineering Society International Conference 2015., ISSN: 1877-7058. DOI: 10.1016/j.proeng.2015.12.577.

[120] H. Shin, Y. Bae, S. Jung, Y. Choi, and C. Kim, "Development of a remote steel pipe cutting robot system," in *2018 15th International Conference on Ubiquitous Robots (UR)*, 2018, pp. 103–107. DOI: 10.1109/URAI.2018.8441772.

[121] K. A. Ghany, H. A. Rafea, and M. Newishy, "Using a Nd:YAG laser and six axes robot to cut zinc-coated steel," *The International Journal of Advanced Manufacturing Technology*, vol. 28, no. 11, pp. 1111–1117, 2006, ISSN: 1433-3015. DOI: 10.1007/s00170-004-2468-x.

[122] I.-G. Park, "Development of 3dimensional precision cutting robot system using plasma cutter for adjustment pipe obtained from shape in the shipyard," in *2021 21st International Conference on Control, Automation and Systems (ICCAS)*, 2021, pp. 2077–2082. DOI: 10.23919/ICCAS52745.2021.9649831.

[123] M. Flemmer, "Plasma-arc cutting control: Investigations into machine vision, modelling and cutting head kinematics," en, Ph.D. dissertation, Massey University, 2018. [Online]. Available: http://hdl.handle.net/10179/15191.

[124] M. Soylak, "A novel real-time control system for plasma cutting robot with xPC target," *Advances in Mechanical Engineering*, vol. 8, no. 4, p. 1 687 814 016 644 128, 2016. DOI: 10.1177/1687814016644128.

[125] H. Wang, G. Ding, Y. Shun, P. Jiang, and G. Yan, "Robot auto-marking and auto-cutting of shipbuilding panels based on a compensation algorithm,"

Industrial Robot: An International Journal, vol. 28, no. 5, pp. 425–434, Jan. 2001, ISSN: 0143-991X. DOI: 10.1108/01439910110401925.

[126] F.-W. Bach, H. Haferkamp, J. Lindemaier, and M. Rachkov, "Underwater climbing robot for contact arc metal drilling and cutting," in *Proceedings of the 1996 IEEE IECON. 22nd International Conference on Industrial Electronics, Control, and Instrumentation*, vol. 3, 1996, pp. 1560–1565. DOI: 10.1109/IECON.1996.570623.

[127] A. S. Dandgawhal, D. S. Sudhakar, and B. T. Patil, "Automated gas cutting machine using open CNC," *IOP Conference Series: Materials Science and Engineering*, vol. 810, no. 1, p. 012 012, Mar. 2020. DOI: 10.1088/1757-899X/810/1/012012.

[128] A. P. Kulkarni, P. Randive, and A. R. Mache, *Micro-Controller Based Oxy-Fuel Profile Cutting System*, version 4797, Nov. 2008. DOI: 10.5281/zenodo.1059871.

[129] Y. He, Y. Xu, Y. Chen, H. Chen, and S. Chen, "Weld seam profile detection and feature point extraction for multi-pass route planning based on visual attention model," *Robotics and Computer-Integrated Manufacturing*, vol. 37, pp. 251–261, 2016, ISSN: 0736-5845. DOI: 10.1016/j.rcim.2015.04.005.

[130] Y. Xu, H. Yu, J. Zhong, T. Lin, and S. Chen, "Real-time seam tracking control technology during welding robot GTAW process based on passive vision sensor," *Journal of Materials Processing Technology*, vol. 212, no. 8, pp. 1654–1662, 2012, ISSN: 0924-0136. DOI: 10.1016/j.jmatprotec.2012.03.007.

[131] X. Gao, D. Ding, T. Bai, and S. Katayama, "Weld-pool image centroid algorithm for seam-tracking vision model in arc-welding process," *IET Image Processing*, vol. 5, pp. 410–419, Aug. 2011, ISSN: 1751-9659. DOI: 10.1049/iet-ipr.2009.0231.

[132] H.-y. Shen, J. Wu, T. Lin, and S.-b. Chen, "Arc welding robot system with seam tracking and weld pool control based on passive vision," *The International Journal of Advanced Manufacturing Technology*, vol. 39, no. 7, pp. 669–678, 2008, ISSN: 1433-3015. DOI: 10.1007/s00170-007-1257-8.

[133] F. Wu, T. F. Flint, K. V. Falch, M. C. Smith, M. Drakopoulos, and W. Mirihanage, "Mapping flow evolution in gas tungsten arc weld pools," *International Journal of Heat and Mass Transfer*, vol. 179, p. 121 679, 2021, ISSN: 0017-9310. DOI: 10.1016/j.ijheatmasstransfer.2021.121679.

[134] J. Liu, Z. Fan, S. I. Olsen, K. H. Christensen, and J. K. Kristensen, "Boosting active contours for weld pool visual tracking in automatic arc welding," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 2, pp. 1096–1108, 2017. DOI: 10.1109/TASE.2015.2498929.

[135] J. J. Wang, T. Lin, and S. B. Chen, "Obtaining weld pool vision information during aluminium alloy TIG welding," *The International Journal of Advanced Manufacturing Technology*, vol. 26, no. 3, pp. 219–227, 2005, ISSN: 1433-3015. DOI: 10.1007/s00170-003-1548-7.

[136] X. Gao and S.-J. Na, "Detection of weld position and seam tracking based on kalman filtering of weld pool images," *Journal of Manufacturing Systems*, vol. 24, no. 1, pp. 1–12, 2005, ISSN: 0278-6125. DOI: 10.1016/S0278-6125(06)00002-1.

[137] K.-Y. Bae, T.-H. Lee, and K.-C. Ahn, "An optical sensing system for seam tracking and weld pool control in gas metal arc welding of steel pipe," *Journal of Materials Processing Technology*, vol. 120, no. 1, pp. 458–465, 2002, ISSN: 0924-0136. DOI: 10.1016/S0924-0136(01)01216-X.

[138] Y. Lu, J. H. M. Lam, and Y. Yam, "Preliminary study on vision-based pen-and-ink drawing by a robotic manipulator," in *2009 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2009, pp. 578–583. DOI: 10.1109/AIM.2009.5229949.

[139] K. Yamazaki, E. Yamamoto, K. Suzuki, *et al.*, "Measurement of surface temperature of weld pools by infrared two colour pyrometry," *Science and Technology of Welding and Joining*, vol. 15, no. 1, pp. 40–47, Jan. 2010. DOI: 10.1179/136217109X12537145658814.

[140] P. Corcoran, P. Mooney, and A. Winstanley, "A convexity measure for open and closed contours," in *Proceedings of the British Machine Vision Conference*, BMVA Press, 2011, pp. 1–11, ISBN: 1-901725-43-X. DOI: 10.5244/C.25.81.

[141] J. Tilley, "Automation, robotics, and the factory of the future," *McKinsey & Company*, Jun. 2017. [Online]. Available: https://www.mckinsey.com/~/media/McKinsey/Business%20Functions/Operations/Our%20Insights/Automation%20robotics%20and%20the%20factory%20of%20the%20future/Automation-robotics-and-the-factory-of-the-future.pdf.

[142] T. Missala, "Paradigms and safety requirements for a new generation of workplace equipment," *International Journal of Occupational Safety and Ergonomics*, vol. 20, no. 2, pp. 249–256, Jan. 2014, ISSN: 1080-3548. DOI: 10.1080/10803548.2014.11077041.

[143] M. Vagaš, D. Šimšik, A. Galajdová, and D. Onofrejová, "Safety as necessary aspect of automated systems," in *2018 16th International Conference on Emerging eLearning Technologies and Applications (ICETA)*, 2018, pp. 617–622. DOI: 10.1109/ICETA.2018.8572130.

[144] J. Akl, S. Kodate, and B. Calli, *Oxy-fuel Cutting Task State (OCTS) image dataset*, version 1.0.0, Zenodo, Mar. 2023. DOI: 10.5281/zenodo.7734951.

[145] M. Hossain, G. Lu, and Y. Yan, "Monitoring and characterisation of oxy-gas burner flames using digital imaging and spectral analysis techniques," in *British Flame Research Technical Meeting on Combustion Diagnostics, Control, Computational Methods & Process Optimisation*, May 2013. DOI: 10.13140/2.1.3697.3126.

[146] J. Smart, G. Lu, Y. Yan, and G. Riley, "Characterisation of an oxy-coal flame through digital imaging," *Combustion and Flame*, vol. 157, no. 6, pp. 1132–1139, 2010, ISSN: 0010-2180. DOI: 10.1016/j.combustflame.2009.10.017.

[147] G. Lu, G. Gilabert, and Y. Yan, "Vision based monitoring and characterisation of combustion flames," *Journal of Physics: Conference Series*, vol. 15, no. 1, p. 194, Jan. 2005. DOI: 10.1088/1742-6596/15/1/033.

[148] R. Sekhar, D. Sharma, and P. Shah, "Intelligent classification of tungsten inert gas welding defects: A transfer learning approach," *Frontiers in Mechanical Engineering*, vol. 8, 2022, ISSN: 2297-3079. DOI: 10.3389/fmech.2022.824038.

[149] Z. Wang, H. Chen, Q. Zhong, *et al.*, "Recognition of penetration state in GTAW based on vision transformer using weld pool image," *The International Journal of Advanced Manufacturing Technology*, vol. 119, no. 7, pp. 5439–5452, 2022, ISSN: 1433-3015. DOI: 10.1007/s00170-021-08538-6.

[150] C. Li, Q. Wang, W. Jiao, M. Johnson, and Y. M. Zhang, "Deep learning-based detection of penetration from weld pool reflection images," *Welding Journal*, vol. 99, no. 9, pp. 239–245, 2020, ISSN: 0043-2296. DOI: 10.29391/2020.99.022.

[151] W. Jiao, Q. Wang, Y. Cheng, R. Yu, and Y. Zhang, "Prediction of weld penetration using dynamic weld pool arc images," *Welding Journal*, vol. 99, pp. 295–302, 2020. DOI: 10.29391/2020.99.027.

[152] Y. Feng, Z. Chen, D. Wang, J. Chen, and Z. Feng, "DeepWelding: A deep learning enhanced approach to GTAW using multisource sensing images," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 1, pp. 465–474, 2020. DOI: 10.1109/TII.2019.2937563.

[153] K. Zhu, W. Chen, Z. Hou, Q. Wang, and H. Chen, *F2GAN based few shot image generation for GMAW defects detection using multi-sensor monitoring system*, 2022. DOI: 10.21203/rs.3.rs-2221109/v1.

[154] M. Rohe, B. N. Stoll, J. Hildebrand, J. Reimann, and J. P. Bergmann, "Detecting process anomalies in the GMAW process by acoustic sensing with a convolutional neural network (CNN) for classification," *Journal of Manufacturing and Materials Processing*, vol. 5, no. 4, 2021, ISSN: 2504-4494. DOI: 10.3390/jmmp5040135.

[155] R. Barot and V. Patel, "Process monitoring and internet of things feasibility for submerged arc welding: State of art," *Materials Today: Proceedings*, vol. 45, pp. 4441–4446, 2021, Second International Conference on Aspects of Materials Science and Engineering (ICAMSE 2021), ISSN: 2214-7853. DOI: 10.1016/j.matpr.2020.12.782.

[156] D. Wu, H. Chen, Y. Huang, and S. Chen, "Online monitoring and model-free adaptive control of weld penetration in VPPAW based on extreme learning machine," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 5, pp. 2732–2740, 2019. DOI: 10.1109/TII.2018.2870933.

[157] S. Kang, M. Kang, Y. H. Jang, and C. Kim, "Deep learning-based penetration depth prediction in Al/Cu laser welding using spectrometer signal and CCD image," *Journal of Laser Applications*, vol. 34, no. 4, p. 042 035, 2022. DOI: 10.2351/7.0000767.

[158] S. Oh, H. Kim, K. Nam, and H. Ki, "Deep-learning approach for predicting laser-beam absorptance in full-penetration laser keyhole welding," *Optics Express*, vol. 29, no. 13, pp. 20 010–20 021, Jun. 2021. DOI: 10.1364/OE.430952.

[159] S. Shevchik, T. Le-Quang, B. Meylan, *et al.*, "Supervised deep learning for real-time quality monitoring of laser welding with X-ray radiographic guidance," *Scientific Reports*, vol. 10, no. 1, p. 3389, 2020, ISSN: 2045-2322. DOI: 10.1038/s41598-020-60294-x.

[160] B. Shen, J. Lu, Y. Wang, *et al.*, "Multimodal-based weld reinforcement monitoring system for wire arc additive manufacturing," *Journal of Materials Research and Technology*, vol. 20, pp. 561–571, 2022, ISSN: 2238-7854. DOI: 10.1016/j.jmrt.2022.07.086.

[161] C. Xia, Z. Pan, Y. Li, J. Chen, and H. Li, "Vision-based melt pool monitoring for wire-arc additive manufacturing using deep learning method," *The International Journal of Advanced Manufacturing Technology*, vol. 120, no. 1, pp. 551–562, 2022, ISSN: 1433-3015. DOI: 10.1007/s00170-022-08811-2.

[162] N. Jamnikar, S. Liu, C. Brice, and X. Zhang, *Comprehensive process-molten pool relations modeling using CNN for wire-feed laser additive manufacturing*, 2021. DOI: 10.48550/ARXIV.2103.11588.

[163] D. K. Maxime, H. N. Raymond, P. Olivier, and B. Tibi, "Anomaly detection in orthogonal metal cutting based on autoencoder method," in *2018 International Conference on Intelligent Systems (IS)*, 2018, pp. 485–493. DOI: 10.1109/IS.2018.8710535.

[164] C. Ajmi, J. Zapata, S. Elferchichi, A. Zaafouri, and K. Laabidi, "Deep learning technology for weld defects classification based on transfer learning and activation features," *Advances in Materials Science and Engineering*, vol. 2020, p. 1 574 350, 2020, ISSN: 1687-8434. DOI: 10.1155/2020/1574350.

[165] M. Sun, M. Yang, B. Wang, L. Qian, and Y. Hong, "Applications of molten pool visual sensing and machine learning in welding quality monitoring," *Journal of Physics: Conference Series*, vol. 2002, no. 1, p. 012 016, Aug. 2021. DOI: 10.1088/1742-6596/2002/1/012016.

[166] J. Rojas, Z. Huang, and K. Harada, "Robot contact task state estimation via position-based action grammars," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, 2016, pp. 249–255. DOI: 10.1109/HUMANOIDS.2016.7803285.

[167] P. Wang, E. Fan, and P. Wang, "Comparative analysis of image classification algorithms based on traditional machine learning and deep learning," *Pattern Recognition Letters*, vol. 141, pp. 61–67, 2021, ISSN: 0167-8655. DOI: 10.1016/j.patrec.2020.07.042.

[168] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein, "On the expressive power of deep neural networks," in *Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y. W. Teh, Eds., ser. Proceedings of Machine Learning Research, vol. 70, PMLR, Aug. 2017, pp. 2847–2854. [Online]. Available: https://proceedings.mlr.press/v70/raghu17a.html.

[169] A. Elhassouny and F. Smarandache, "Trends in deep convolutional neural networks architectures: A review," in *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*, 2019, pp. 1–8. DOI: 10.1109/ICCSRE.2019.8807741.

[170] S. Sun, Z. Cao, H. Zhu, and J. Zhao, "A survey of optimization methods from a machine learning perspective," *IEEE Transactions on Cybernetics*, vol. 50, no. 8, pp. 3668–3681, 2020. DOI: 10.1109/TCYB.2019.2950779.

[171] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference for Learning Representations*, 2015. DOI: 10.48550/ARXIV.1412.6980.

[172] W. R. Stahel, "The circular economy," *Nature*, vol. 531, no. 7595, pp. 435–438, 2016, ISSN: 1476-4687. DOI: 10.1038/531435a.

[173] M. Geissdoerfer, P. Savaget, N. M. Bocken, and E. J. Hultink, "The circular economy – a new sustainability paradigm?" *Journal of Cleaner Production*, vol. 143, pp. 757–768, 2017, ISSN: 0959-6526. DOI: 10.1016/j.jclepro.2016.12.048.

[174] J. S. Ribeiro and J. de Oliveira Gomes, "Proposed framework for end-of-life aircraft recycling," *Procedia CIRP*, vol. 26, pp. 311–316, 2015, 12th Global Conference on Sustainable Manufacturing – Emerging Potentials, ISSN: 2212-8271. DOI: 10.1016/j.procir.2014.07.048.

[175] M. Sabaghi, Y. Cai, C. Mascle, and P. Baptiste, "Sustainability assessment of dismantling strategies for end-of-life aircraft recycling," *Resources, Conservation and Recycling*, vol. 102, pp. 163–169, 2015, ISSN: 0921-3449. DOI: 10.1016/j.resconrec.2015.08.005.

[176] F. C. Rios, W. K. Chong, and D. Grau, "Design for disassembly and deconstruction – challenges and opportunities," *Procedia Engineering*, vol. 118, pp. 1296–1304, 2015, Defining the future of sustainability and resilience in design, engineering and construction, ISSN: 1877-7058. DOI: 10.1016/j.proeng.2015.08.485.

[177] N. A. W. A. Zawawi, M. S. Liew, and K. L. Na, "Decommissioning of offshore platform: A sustainable framework," in *2012 IEEE Colloquium on Humanities, Science and Engineering (CHUSER)*, 2012, pp. 26–31. DOI: 10.1109/CHUSER.2012.6504275.

[178] H. Yan, L. Wu, and J. Yu, "The environmental impact analysis of hazardous materials and the development of green technology in the shipbreaking process," *Ocean Engineering*, vol. 161, pp. 187–194, 2018, ISSN: 0029-8018. DOI: 10.1016/j.oceaneng.2018.03.024.

[179] J.-K. Choi, D. Kelley, S. Murphy, and D. Thangamani, "Economic and environmental perspectives of end-of-life ship management," *Resources, Conservation and Recycling*, vol. 107, pp. 82–91, 2016, ISSN: 0921-3449. DOI: 10.1016/j.resconrec.2015.12.007.

[180] S. M. Rahman and J. Kim, "Circular economy, proximity, and shipbreaking: A material flow and environmental impact analysis," *Journal of Cleaner Production*, vol. 259, p. 120 681, 2020, ISSN: 0959-6526. DOI: 10.1016/j.jclepro.2020.120681.

[181] S. Lee, W. Pan, T. Linner, and T. Bock, "A framework for robot assisted deconstruction: Process, sub-systems and modelling," in *32nd ISARC: Proceedings of the International Symposium on Automation and Robotics in Construction*, 2015. [Online]. Available: https://mediatum.ub.tum.de/1484209.

[182] X. Wei, S. Qiu, L. Zhu, *et al.*, "Toward support-free 3D printing: A skeletal approach for partitioning models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 10, pp. 2799–2812, 2018. DOI: 10.1109/TVCG.2017.2767047.

[183] L. Luo, I. Baran, S. Rusinkiewicz, and W. Matusik, "Chopper: Partitioning models into 3D-printable parts," *ACM Transactions on Graphics*, vol. 31, no. 6, Nov. 2012, ISSN: 0730-0301. DOI: 10.1145/2366145.2366148.

[184] S. C. Stein, M. Schoeler, J. Papon, and F. Worgotter, "Object partitioning using local convexity," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2014.

[185] H. Poschmann, H. Brüggemann, and D. Goldmann, "Disassembly 4.0: A review on using robotics in disassembly tasks as a way of automation," *Chemie Ingenieur Technik*, vol. 92, no. 4, pp. 341–359, 2020. DOI: 10.1002/cite.201900107.

[186] F. Pistolesi and B. Lazzerini, "TeMA: A tensorial memetic algorithm for many-objective parallel disassembly sequence planning in product refurbishment," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3743–3753, 2019. DOI: 10.1109/TII.2019.2904631.

[187] Y.-S. Ma, H.-B. Jun, H.-W. Kim, and D.-H. Lee, "Disassembly process planning algorithms for end-of-life product recovery and environmentally conscious disposal," *International Journal of Production Research*, vol. 49, no. 23, pp. 7007–7027, Dec. 2011, ISSN: 0020-7543. DOI: 10.1080/00207543.2010.495089.

[188] H.-J. Han, J.-M. Yu, and D.-H. Lee, "Mathematical model and solution algorithms for selective disassembly sequencing with multiple target components and sequence-dependent setups," *International Journal of Production Research*, vol. 51, no. 16, pp. 4997–5010, Aug. 2013, ISSN: 0020-7543. DOI: 10.1080/00207543.2013.788794.

[189] M.-L. Lee, S. Behdad, X. Liang, and M. Zheng, "Disassembly sequence planning considering human-robot collaboration," in *2020 American Control Conference (ACC)*, 2020, pp. 2438–2443. DOI: 10.23919/ACC45564.2020.9147652.

[190] W. Xu, Q. Tang, J. Liu, Z. Liu, Z. Zhou, and D. T. Pham, "Disassembly sequence planning using discrete bees algorithm for human-robot collaboration in remanufacturing," *Robotics and Computer-Integrated Manufacturing*, vol. 62, p. 101 860, 2020, ISSN: 0736-5845. DOI: 10.1016/j.rcim.2019.101860.

[191] Y. Laili, X. Li, Y. Wang, L. Ren, and X. Wang, "Robotic disassembly sequence planning with backup actions," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 2095–2107, 2022. DOI: 10.1109/TASE.2021.3072663.

[192] M. Kheder, M. Trigui, and N. Aifaoui, “Optimization of disassembly sequence planning for preventive maintenance,” *The International Journal of Advanced Manufacturing Technology*, vol. 90, no. 5, pp. 1337–1349, 2017, ISSN: 1433-3015. DOI: 10.1007/s00170-016-9434-2.

[193] F. Giudice and G. Fargione, “Disassembly planning of mechanical systems for service and recovery: A genetic algorithms based approach,” *Journal of Intelligent Manufacturing*, vol. 18, no. 3, pp. 313–329, 2007, ISSN: 1572-8145. DOI: 10.1007/s10845-007-0025-9.

[194] K.-K. Seo, J.-H. Park, and D.-S. Jang, “Optimal disassembly sequence using genetic algorithms considering economic and environmental aspects,” *The International Journal of Advanced Manufacturing Technology*, vol. 18, no. 5, pp. 371–380, 2001, ISSN: 1433-3015. DOI: 10.1007/s001700170061.

[195] L. M. Galantucci, G. Percoco, and R. Spina, “Assembly and disassembly planning by using fuzzy logic & genetic algorithms,” *International Journal of Advanced Robotic Systems*, vol. 1, no. 2, p. 7, 2004. DOI: 10.5772/5622.

[196] S. Vongbunyong, S. Kara, and M. Pagnucco, “A framework for using cognitive robotics in disassembly automation,” in *Leveraging Technology for a Sustainable World*, D. A. Dornfeld and B. S. Linke, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 173–178, ISBN: 978-3-642-29069-5.

[197] X. Guo, S. Liu, M. Zhou, and G. Tian, “Disassembly sequence optimization for large-scale products with multiresource constraints using scatter search and Petri nets,” *IEEE Transactions on Cybernetics*, vol. 46, no. 11, pp. 2435–2446, 2016. DOI: 10.1109/TCYB.2015.2478486.

[198] Y. Tang, M. Zhou, and R. Caudill, “An integrated approach to disassembly planning and demanufacturing operation,” *IEEE Transactions on Robotics and Automation*, vol. 17, no. 6, pp. 773–784, 2001. DOI: 10.1109/70.975899.

[199] Y. Tang, “Learning-based disassembly process planner for uncertainty management,” *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, vol. 39, no. 1, pp. 134–143, 2009. DOI: 10.1109/TSMCA.2008.2007990.

[200] C. He, Z. Jin, R. Gu, and H. Qu, “Automatic disassembly and recovery device for mobile phone circuit board CPU based on machine vision,” *Journal of Physics: Conference Series*, vol. 1684, no. 1, p. 012 137, Nov. 2020. DOI: 10.1088/1742-6596/1684/1/012137.

[201] A. ElSayed, E. Kongar, S. M. Gupta, and T. Sobh, “A robotic-driven disassembly sequence generator for end-of-life electronic products,” *Journal of Intelligent & Robotic Systems*, vol. 68, no. 1, pp. 43–52, 2012, ISSN: 1573-0409. DOI: 10.1007/s10846-012-9667-8.

[202] M. Merdan, W. Lepuschitz, T. Meurer, and M. Vincze, “Towards ontology-based automated disassembly systems,” in *IECON 2010 – 36th Annual Conference on IEEE Industrial Electronics Society*, 2010, pp. 1392–1397. DOI: 10.1109/IECON.2010.5675479.

[203] A. S. Allam and M. Nik-Bakht, “From demolition to deconstruction of the built environment: A synthesis of the literature,” *Journal of Building Engineering*, vol. 64, p. 105 679, 2023, ISSN: 2352-7102. DOI: 10.1016/j.jobe.2022.105679.

[204] M. Helal and T. Bock, “Towards a sustainable construction industry: Automated deconstruction and recycling,” *Democratic Transition and Sustainable Communities*, p. 611, 2013. [Online]. Available: https://www.researchgate.net/profile/Ayman-Othman/publication/271906673_SB13_Cairo_Proceedings/links/54d5f7ee0cf246475808cb53/SB13-Cairo-Proceedings.pdf#page=625.

[205] B. Sanchez, C. Rausch, C. Haas, and R. Saari, "Multi-objective optimization analysis for selective disassembly planning of buildings," in *Proceedings of the 36th International Symposium on Automation and Robotics in Construction (ISARC)*, M. Al-Hussein, Ed., Banff, Canada: International Association for Automation and Robotics in Construction (IAARC), May 2019, pp. 128–135, ISBN: 978-952-69524-0-6. DOI: 10.22260/ISARC2019/0018.

[206] E. Queheille, F. Taillandier, and N. Saiyouri, "Optimization of strategy planning for building deconstruction," *Automation in Construction*, vol. 98, pp. 236–247, 2019, ISSN: 0926-5805. DOI: 10.1016/j.autcon.2018.11.007.

[207] C. Kasperzyk, M.-K. Kim, and I. Brilakis, "Automated re-prefabrication system for buildings using robotics," *Automation in Construction*, vol. 83, pp. 184–195, 2017, ISSN: 0926-5805. DOI: 10.1016/j.autcon.2017.08.002.

[208] E. Lublasser, L. Hildebrand, A. Vollpracht, and S. Brell-Cokcan, "Robot assisted deconstruction of multi-layered façade constructions on the example of external thermal insulation composite systems," *Construction Robotics*, vol. 1, no. 1, pp. 39–47, 2017, ISSN: 2509-8780. DOI: 10.1007/s41693-017-0001-7.

[209] E. Lublasser, K. Iturralde, T. Linner, S. Brell Cokcan, and T. Bock, "Automated refurbishment & end-of-life processes research approaches in German and Japanese construction.," in *Proceedings of the CIB*IAARC W119 CIC 2016 Workshop*, 2016. [Online]. Available: https://mediatum.ub.tum.de/1484220.

[210] B. Sanchez, C. Rausch, C. Haas, and T. Hartmann, "A framework for BIM-based disassembly models to support reuse of building components," *Resources, Conservation and Recycling*, vol. 175, p. 105 825, 2021, ISSN: 0921-3449. DOI: 10.1016/j.resconrec.2021.105825.

[211] B. Sanchez, C. Rausch, and C. Haas, "Deconstruction programming for adaptive reuse of buildings," *Automation in Construction*, vol. 107, p. 102 921, 2019, ISSN: 0926-5805. DOI: 10.1016/j.autcon.2019.102921.

[212] B. Sanchez, C. Rausch, C. Haas, and T. Hartmann, "VPL prototypes for determining disassembly parameters of BIM models," in *Proceedings of 2021 4th International Conference on Civil Engineering and Architecture*, T. Kang and Y. Lee, Eds., Singapore: Springer Nature Singapore, 2022, pp. 507–514, ISBN: 978-981-16-6932-3.

[213] C. Rausch, B. Sanchez, and C. Haas, "Spatial parameterization of non-semantic CAD elements for supporting automated disassembly planning," *Modular and Offsite Construction (MOC) Summit Proceedings*, pp. 108–115, 2019. DOI: 10.29173/mocs83.

[214] R. Volk, T. H. Luu, J. S. Mueller-Roemer, N. Sevilmis, and F. Schultmann, "Deconstruction project planning of existing buildings based on automated acquisition and reconstruction of building information," *Automation in Construction*, vol. 91, pp. 226–245, 2018, ISSN: 0926-5805. DOI: 10.1016/j.autcon.2018.03.017.

[215] X. J. Ge, P. Livesey, J. Wang, S. Huang, X. He, and C. Zhang, "Deconstruction waste management through 3D reconstruction and BIM: A case study," *Visualization in Engineering*, vol. 5, no. 1, p. 13, 2017, ISSN: 2213-7459. DOI: 10.1186/s40327-017-0050-5.

[216] Y. Wei, A. Pushkar, and B. Akinci, "Supporting deconstruction waste management through 3D imaging: A case study," in *Proceedings of the 36th International Symposium on Automation and Robotics in Construction (ISARC)*, M. Al-Hussein, Ed., Banff, Canada: International Association for Automation and Robotics in Construction (IAARC), May 2019, pp. 438–445, ISBN: 978-952-69524-0-6. DOI: 10.22260/ISARC2019/0059.

[217] P. A. E. Brey, "Anticipatory ethics for emerging technologies," *NanoEthics*, vol. 6, no. 1, pp. 1–13, 2012, ISSN: 1871-4765. DOI: 10.1007/s11569-012-0141-7.

[218] J. C. Mankins, "Technology readiness assessments: A retrospective," *Acta Astronautica*, vol. 65, no. 9, pp. 1216–1223, 2009, ISSN: 0094-5765. DOI: 10.1016/j.actaastro.2009.03.058.

[219] United States General Accounting Office, *Federal surplus ships: Government efforts to address the growing backlog of ships awaiting disposal, ADA355139*, Oct. 1998. [Online]. Available: https://apps.dtic.mil/sti/pdfs/ADA355139.pdf.

[220] N. Mikelis, "Ship recycling," in *Sustainable Shipping: A Cross-Disciplinary View*, H. N. Psaraftis, Ed., Cham: Springer International Publishing, 2019, pp. 203–248. DOI: 10.1007/978-3-030-04330-8_6.

[221] International Labour Organization (ILO), *Ship-breaking: A hazardous work*, Mar. 2015. [Online]. Available: https://www.ilo.org/safework/areasofwork/hazardous-work/WCMS_356543/lang--en/index.htm.

[222] NGO Shipbreaking Platform, *The toxic tide: 2022 shipbreaking records*, 2023. [Online]. Available: https://www.offthebeach.org/.

[223] D. Cords and K. Prettner, "Technological unemployment revisited: automation in a search and matching framework," *Oxford Economic Papers*, vol. 74, no. 1, pp. 115–135, Jun. 2021, ISSN: 0030-7653. DOI: 10.1093/oep/gpab022.

[224] M. Beane, "Shadow learning: Building robotic surgical skill when approved means fail," *Administrative Science Quarterly*, vol. 64, no. 1, pp. 87–123, Jan. 2018, ISSN: 0001-8392. DOI: 10.1177/0001839217751692.

[225] M. Beane, *How do we learn to work with intelligent machines?* TED, Nov. 2018. [Online]. Available: https://www.ted.com/talks/matt_beane_how_do_we_learn_to_work_with_intelligent_machines.

[226] United States Census Bureau, *QuickFacts: Brownsville city, Texas*, Jul. 2022. [Online]. Available: https://www.census.gov/quickfacts/fact/table/brownsvillecitytexas.

[227] Z. Yang, Z. Jiang, T. Notteboom, and H. Haralambides, "The impact of ship scrapping subsidies on fleet renewal decisions in dry bulk shipping," *Transportation Research Part E: Logistics and Transportation Review*, vol. 126, pp. 177–189, 2019, ISSN: 1366-5545. DOI: 10.1016/j.tre.2019.04.008.

[228] F. W. Wolek, "The skill development processes of apprenticeship," *International Journal of Lifelong Education*, vol. 18, no. 5, pp. 395–406, Sep. 1999, ISSN: 0260-1370. DOI: 10.1080/026013799293630.

[229] S. R. Epstein, "Craft guilds, apprenticeship, and technological change in preindustrial europe," *The Journal of Economic History*, vol. 58, no. 3, pp. 684–713, 1998. DOI: 10.1017/S0022050700021124.

[230] F. G. Agyemang and H. Boateng, "Tacit knowledge transfer from a master to an apprentice among hairdressers," *Education + Training*, vol. 61, no. 1, pp. 108–120, Jan. 2019, ISSN: 0040-0912. DOI: 10.1108/ET-12-2017-0200.

[231] E. Gray and K. G. Farrell, "Perceptions of culinary education and chef apprenticeships in ireland," *Higher Education, Skills and Work-Based Learning*, vol. 11, no. 1, pp. 228–240, Jan. 2021, ISSN: 2042-3896. DOI: 10.1108/HESWBL-06-2019-0083.

[232] M. Stierand, V. Dörfler, and P. Lynch, "Haute cuisine innovations: The role of the master-apprentice relationship," English, in *British Academy of Management Annual Conference*, Sep. 2008. [Online]. Available: https://pure.buas.nl/

en/publications/haute-cuisine-innovations-the-role-of-the-master-apprentice-relat.

[233] D. Belcher, "The apprenticeship approach to advanced academic literacy: Graduate students and their mentors," *English for Specific Purposes*, vol. 13, no. 1, pp. 23–34, 1994, ISSN: 0889-4906. DOI: 10.1016/0889-4906(94)90022-1.

[234] F. W. Wolek and J. W. Klingler, "Apprenticeship and the transfer of technical know-how," *The Journal of Technology Transfer*, vol. 23, no. 3, pp. 51–57, 1998, ISSN: 1573-7047. DOI: 10.1007/BF02509575.

[235] J. Hsuan and C. Parisi, "Mapping the supply chain of ship recycling," *Marine Policy*, vol. 118, p. 103 979, 2020, ISSN: 0308-597X. DOI: 10.1016/j.marpol.2020.103979.

[236] M. Sarraf, F. Stuer-Lauridsen, M. Dyoulgerov, R. Bloch, S. Wingfield, and R. Watkinson, *The ship breaking and recycling industry in Bangladesh and Pakistan*, The World Bank, 2010. DOI: 10.11588/xarep.00003749.

[237] C. Spinuzzi, "The methodology of participatory design," *Technical Communication*, vol. 52, no. 2, pp. 163–174, 2005, ISSN: 0049-3155. [Online]. Available: https://www.ingentaconnect.com/content/stc/tc/2005/00000052/00000002/art00005.

[238] W. A. Rogers, T. Kadylak, and M. A. Bayles, "Maximizing the benefits of participatory design for human–robot interaction research with older adults," *Human Factors*, vol. 64, no. 3, pp. 441–450, 2022. DOI: 10.1177/00187208211037465.

[239] K. Winkle, E. Senft, and S. Lemaignan, "LEADOR: A method for end-to-end participatory design of autonomous social robots," *Frontiers in Robotics and AI*, vol. 8, 2021, ISSN: 2296-9144. DOI: 10.3389/frobt.2021.704119.

[240] S. Amoroso, P. Moncada-Paternò-Castello, and A. Vezzani, "R&D profitability: The role of risk and Knightian uncertainty," *Small Business Economics*, vol. 48, no. 2, pp. 331–343, 2017, ISSN: 1573-0913. DOI: 10.1007/s11187-016-9776-z.

[241] N. Kolt, "Algorithmic black swans," *Washington University Law Review*, vol. 101, 2023. [Online]. Available: https://ssrn.com/abstract=4370566.

[242] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, "Explainable AI: A review of machine learning interpretability methods," *Entropy*, vol. 23, no. 1, 2021, ISSN: 1099-4300. DOI: 10.3390/e23010018.

[243] G. Downey, "Are engineers losing control of technology?: From 'problem solving' to 'problem definition and solution' in engineering education," *Chemical Engineering Research and Design*, vol. 83, no. 6, pp. 583–595, 2005, ISSN: 0263-8762. DOI: 10.1205/cherd.05095.

[244] A. Birk, "What is robotics? an interdisciplinary field is getting even more diverse," *IEEE Robotics & Automation Magazine*, vol. 18, no. 4, pp. 94–95, 2011. DOI: 10.1109/MRA.2011.943235.

[245] K. Mainzer, "Interdisciplinarity and innovation dynamics. on convergence of research, technology, economy, and society," *Poiesis & Praxis*, vol. 7, no. 4, pp. 275–289, 2011, ISSN: 1615-6617. DOI: 10.1007/s10202-011-0088-8.

[246] T. Kose and I. Sakata, "Identifying technology convergence in the field of robotics research," *Technological Forecasting and Social Change*, vol. 146, pp. 751–766, 2019, ISSN: 0040-1625. DOI: 10.1016/j.techfore.2018.09.005.

[247] O. Michalec, C. O'Donovan, and M. Sobhani, "What is robotics made of? the interdisciplinary politics of robotics research," *Humanities and Social Sciences Communications*, vol. 8, no. 1, p. 65, 2021, ISSN: 2662-9992. DOI: 10.1057/s41599-021-00737-6.