OUKA
The University of Osaka
Institutional Knowledge Archive

Title	Design, Evaluation and Deployment of Gestural Interfaces: From Multicultural Perception to Accessible Multi-Scale Interactions
Author(s)	Soares Medeiros, Anna Carolina
Citation	大阪大学, 2021, 博士論文
Version Type	VoR
URL	https://doi.org/10.18910/85270
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

The University of Osaka

Design, Evaluation and Deployment of Gestural Interfaces: From Multicultural Perception to Accessible Multi-Scale Interactions

Submitted to
Graduate School of Information
Science and Technology
Osaka University

April 2021

Anna Carolina SOARES MEDEIROS

Thesis Committee

Prof. Haruo Takemura (Osaka University)
Prof. Tatsuhiro Tsuchiya (Osaka University)
Prof. Masanori Hashimoto (Osaka University)
Assoc. Prof. Yuki Uranishi (Osaka University)

List of Publications

Peer-Reviewed Journals

1. Medeiros, A. C. S., Ratsamee, P., Orlosky, J., Uranishi, Y., Higashida, M., Takemura, H. (2020). 3D Pointing Gestures as Target Selection Tools for Monocular UAVs. In Robotic and Mechatronics Journal.

Peer-Reviewed Conference Proceedings

1. Medeiros, A. C. S., Ratsamee, P., Orlosky, J., Uranishi, Y., Higashida, M., Takemura, H. (2021). UAV Target-Selection: 3D Pointing Interface System for Large-Scale Environment. In International Conference on Robotics and Automation (ICRA2021)

2. Medeiros, A. C. S., Ratsamee, P., Uranishi, Y., Takemura, H.(2020). Human-Drone Interaction: Using Pointing Gesture to Define a Target Object. *International Conference on Human-Computer Interaction (HCII2020))*.

3. Medeiros, A. C. S., Ratsamee, P., Uranishi, Y., Mashita, T., Takemura, H.(2019). 3D Gesture Interface: Japan-Brazil Perceptions. *Proceedings of the International Conference on Human-Computer Interaction (HCII2019))*, (pp. 266-279). doi:10.1007/978-3-030-22577-3_19

4. Medeiros, A. C. S., Tavares, T. A. and da Fonseca, I. E. (2015). How to Design an User Interface Based on Gestures?. *Proceedings of International Conference of Design, User Experience, and Usability*, (pp. 63-74.). doi:10.1007/978-3-319-20886-2_7

5. Medeiros, A. C. S., Tavares, T. A. and da Fonseca, I. E. (2014). Development Process of User Interfaces Based on Gestures. *Companion Proceedings of the 20th Brazilian Symposium on Multimedia and Web (WEBMEDIA2014)*, (pp. 13-21). ISSN 2596-1683

Domestic Workshops

1. Medeiros, A. C. S., Ratsamee, P., Uranishi, Y., Mashita, T., Takemura, H., Tavares, T. A. (2017). Mouse vs Gesture：日本におけるインタフェースの知覚. *SIG-MR (*複合現実感研究会*)*. Japan.

2. Castro, R. H. A. de, Medeiros, A. C. S., Santos, F. G. dos, Tavares, T. A., Fonseca, I. E. da. (2014). The Use of Natural Interaction in Three-dimensional Environments for Training in the Electrical Sector. *Workshop of Virtual and Augmented Reality Applied to Education (CBIE, 2014)*. Vol. 3, No. 1, pp.435. Brazil. doi:10.5753/cbie.wcbie.2014.495

3. Medeiros, A. C. S.,Castro, R. H. A. de, Silva, D. R. D. da, Tavares, T. A. (2012). Natural Interaction as Support for Distance Mentorship Activity in Telemedicine. *Immersive Environments Workshop for Education (WAIEd 2012)*. doi:10.22456/1679-1916.36128

4. Castro, R. H. A. de, Medeiros, A. C. S., Tavares, T. A. (2012). Natural Interaction for 3D Manipulation in Telemedicine: A Study Case Developed for Arthron Video Collaboration Tool. *Workshop of Virtual and Augmented Reality (WRVA2012)*. Brazil.

International Conference Poster Presentation

1. Tavares, T. A., Castro, R. H. A., Medeiros, A. C. S. and Anjos, E. (2013). The Use of Natural Interaction to Enrich the User Experience in Telemedicine Systems *International Conference on Human-Computer Interaction (HCII2013)*. Posters Extended Abstracts, pp. 220-224. doi:10.1007/978-3-642-39476-8_46

Abstract

Gestural Interface uses body movements to give commands to a computational system; it can complement or substitute navigation in a conventional interface. The present work explores Gestural Interfaces from the conceptual development process of choosing gestures to a system function until the application and implementation of gestures in short and long distances settings; here referred to as "multi-scaled" interaction. A short-scaled gesture interface was analyzed by users in Brazil and Japan, whereas the multi-scaled gesture interface was developed and studied in Japan.

The Gesture Development Process presented here provides a flexible solution for system functions with varying levels of complexity and enables collaboration and reuse of past efforts. To illustrate the Gesture Development Process, gestures for the standard system functions "Select," "Rotate," "Translate," "Scale," and "Stop" were developed.

Analyzing how gesture interfaces are perceived in different parts of the world can help in the reuse of gestural interfaces created for a specific culture and help understand possible weak and strong points, allowing for improvements to the interfaces. This study also aims to analyze how different technological familiarization levels impact gestural interfaces' user experience. This work describes the findings of an experiment replicated in two countries: Brazil and Japan. In each experiment, 20 subjects tested two applications; one had a mouse-based interface, one had a gesture-based interface. User experience was measured using questionnaires. The gestural interface was perceived similarly in both countries. This similarity is a cue for reusing the gestures developed for one user-group with the other.

Multi-scaled interaction scenarios include short-scaled interaction where user, camera, and interaction targets are all close together (within 3 meters), and long-scaled interaction scenarios, where the distances between the user, camera, and targets provide far more challenges. For example, when pointing to targets far away, the slightest change in the pointing line will significantly increase the final destination.

This thesis investigates multi-scaled interaction scenarios. The author used a drone to provide accessibility outdoors, where the Firefighting scenario needed a long distance between user-drone and user-target. For this purpose, the author conducted a Preliminary Interview and User Study with seven subjects from the Kobe Firefighting Brigade, Japan. As a drone's flight and locomotion properties significantly affect the user's mental and physical expectations differently than other grounded robots, a careful investigation of user-defined design preferences and interactions is required. This work proposes an examination and discussion, with experienced firefighters, about Human-Drone Interactions when relying solely on the drone's monocular camera, without relying on other devices such as GPS or external cameras. The User Study had three main elements: A drone, a building, and the volunteering firefighters. During the study, each firefighter should specify a window to the drone. The drone would theoretically use that window to enter the building and perform some designed task, like information gathering, thus saving the firefighter time and effort. Results show that the subjects always chose Pointing Gestures and voice commands to communicate to the drone a target window.

The author first built a 2D gesture interface application with the resulting gesture from this investigation. Since that 2D interface failed to show satisfactory results, a 3D gesture interface application was proposed. The author includes a description of the process for detecting and localizing objects and carrying out the transformations between coordinate systems. The proposed 3D pointing gesture interface improves the previous 2D pointing gesture interface by integrating depth information with Simultaneous Localization and Mapping (SLAM) Techniques, solving multiple objects aligned on the same plane ambiguity in a large-scale outdoor environment. Experimental results showed that the 3D pointing gesture interface obtained a 0.85 and 0.73 F1-Score on average in simulation and real-world experiments and 0.58 F1-Score at the maximum distance of 25 meters between drone and building.

Keywords: Gesture Interaction, Human-Computer Interaction, Human-Drone Interaction, Intention Recognition, Multi-Scale Interaction.

Acknowledgements

There are many who helped me along the way on this journey. I want to take a moment to thank them.

First, I wish to thank my dissertation committee. Without their guidance, I would not have made it. I would like to thank Prof. Haruo Takemura, my supervisor, whose comments and continuous support and encouragement steered me through this research. Also, Assoc. Prof. Yuki Uranishi for the support and comments, Prof. Tatsuhiro Tsuchiya and Prof. Masanori Hashimoto served as wise committee members, and reviewed my thesis.

I would like to express my sincere gratitude and appreciation for Assist. Prof. Photchara Ratsamee whose insight, guidance, support, encouragement, and knowledge into the subject matter has been invaluable throughout this study. I am specially grateful for the opportunity he gave me to work on the MBZIRC 2020 robotics challenge. I would like to thank Assoc. Prof. Jason Orlosky for his continued support throughout this project, and Assoc. Prof. Tomohiro Mashita and Assoc. Prof. Manabu Higashida for the valuable comments.

I am forever grateful for all present and past members of Takemura Lab., including Assoc. Prof. Shizuka Shirai, Ms. Etsu Sakaki, and Prof. Kiyoshi Kiyokawa, who have taken their valuable time to give feedback at some point of my work, or just helped me with Japanese, thank you.

I would like to thank Kobe Fire Academy for their cooperation and collaboration in the course of this work.

To my parents, my sister, and my friends: you put up with me being distracted and missing many events. I am forever grateful for your patience and understanding. I hope to have time now to reconnect with each of you.

Finally, to my husband, Rafael: your love and understanding helped me through the dark times. Without you believing in me, I never would have made it. It is time to celebrate; you earned this degree right along with me.

Anna Carolina Soares Medeiros

Osaka University

January 2021

Contents

1 Introduction **1**

1.1 Natural Interaction 1
1.2 The Importance of the Process of Gesture Development 2
1.3 Multi-scale Gestural Interactions 2
1.4 Statement of the Problem 3
1.5 Research Questions 3
1.6 Philosophy 4
1.7 Outline of the Dissertation 6

2 Literature Review **8**

2.1 Gesture Development Process 8
2.1.1 Comparative Analysis 18
2.2 Multicultural Perceptions 19
2.3 Multi-Scale Gesture Interface and UAVs 19

3 Gesture Development Process **23**

3.1 Overview 23
3.1.1 Step 1: Define Functions 23
3.1.2 Step 2: Generate Gestures 24
3.1.3 Third Step: Evaluation 25
3.1.4 Artifacts 27
3.2 Validating the Gesture Development Process in the Three-dimensional Navigation Scenario 28
3.2.1 First Step 28
3.2.2 Second Step 29

3.2.3 Third Stage 31
3.2.4 Development of CubeNI 35
3.2.5 Evaluation of User Experience 39
3.3 Conclusion 44
3.3.1 Contributions 45

4 Multicultural Gestural Perceptions 47
4.1 Introduction 47
4.2 Overview of experiment 47
4.3 Materials and Human Resources 49
4.3.1 Participants 49
4.3.2 Applications 49
4.3.3 Tangible Resources 50
4.3.4 Questionnaires 51
4.4 Conducting the Experiment 51
4.4.1 Routine 52
4.4.2 Results 52
4.5 Discussion 59
4.6 Conclusion 60
4.6.1 Contributions 62

5 Accessible Multi-Scale Gestural Interaction 63
5.1 Introduction 63
5.2 Elicitation Study 66
5.2.1 Environment 66
5.2.2 Results 67
5.3 2D Pointing Gesture Interface Approach 69
5.4 3D Pointing Gesture Interface Approach 79
5.4.1 Calibration 81
5.4.2 Depth Estimation 82
5.4.3 3D Target Estimation 82
5.5 Experiments and Results 84

5.5.1 Simulation Experiment 84
5.5.2 Real-World Experiment 90
5.5.3 Comparing Simulation and Real-World Experiment Conditions 96
5.6 Discussion 96
5.7 Conclusion 100
5.7.1 Contributions 101

6 Conclusion **102**
6.1 Summary of Major Findings 102

List of Tables

2.1 Comparison of related works on gesture development process . . 19
2.2 Comparison of related works on gesture interface for UAVs. Short-scale are for distances less than 2 meters. The author tested large-scale until 25 meters. 22

3.1 Attribution of Semantics Test Results 32
3.2 Memory Test Results 33
3.3 Stress Test Results on Gesture Vocabulary 1 34
3.4 Stress Test Results on Gesture Vocabulary 2 34

5.1 Summary of Experiments on Simulation Environment 88
5.2 Summary of Experiments on Real-World Environment 93

List of Figures

1.1 Overview of this dissertation 7

2.1 Example of technology based gestures. 9
2.2 Human based process defined in Nielsen et al. 9
2.3 Step D evaluates the Gesture Vocabulary using three tests. Each test evaluate different attributes. 10
2.4 Scenario I from stationary camera showing the customer (left), and from head-mounted camera (right) showing the bartender . . 11
2.5 Scenario II-III from stationary camera (left) and head-mounted camera (right) 12
2.6 Henze et al. gesture development process was based on Nielsen et al. 13
2.7 Exemplary camera image from the participatory design study by Henze et al. 13
2.8 A static gesture can be defined as a specific position of the hands or body. A dynamic gesture changes the position over time to complete the gesture. A hybrid gesture might be static on one hand and dynamic in the other. 14
2.9 The set of static gestures (from a to g: Start, pause, stop, increase volume, decrease volume, next, and previous) 15
2.10 The set of dynamic gestures (from left to right: Start/pause, stop, increase volume, and next) 15
2.11 Demo system for the human to TV interaction 17
2.12 Wu et al. design guidelines steps 18

3.1 Proposed Gesture Development Process 24
3.2 Generating gestures from elicitation studies in a cyclic manner . 25
3.3 Proposed Gesture Development Process on the 3D Navigation Scenario 28
3.4 CubeMK 29
3.5 Gestures generated for the “Select” function. 30
3.6 Gestures generated for the “Rotate” function. 30
3.7 Gestures generated for the “Translate” function. 31
3.8 Gestures generated for the “Scale” function. 32
3.9 Gestures generated for the “Stop” function. 33
3.10 Gesture R7 34
3.11 Gesture Vocabulary 1 35
3.12 Gesture Vocabulary 2 36
3.13 Template of Final Gestures 37
3.14 UML use case diagram of CUBENI 38
3.15 Image of the Cube that suffers the selection, rotation, translation, scaling and stop actions. The same Cube is used in two applications with mouse and keyboard (CubeMK) and gestural (CubeNI) interfaces. 41
3.16 Comparison between CubeMK and CubeNI using pairs of words from AttrakDiff 42
3.17 Average results 43
3.18 View of the final test results 44

4.1 CubeNI’s Gestures 50
4.2 CubeNI and CubeMK’s possible interactions with the red cube. . 50
4.3 Experiment set-up 51
4.4 Experiment’s Routine 52
4.5 Brazilian Experiment’s Diagram of Average Values 53
4.6 Japanese Experiment’s Diagram of Average Values 53
4.7 Brazil’s Description of Word-Pairs 55
4.8 Japan’s Description of Word-Pairs 55

4.9 Brazil's Portfolio of Results 56
4.10 Japan's Portfolio of Results 56
4.11 Student's Paired T-test overall results between Brazil and Japan. 57
4.12 Student's Paired T-test results between Brazil and Japan, considering every possible comparison regarding the four qualities (PQ, HQ-I, HQ-S and ATT). 58

5.1 Possible scenario for the system presented here 65
5.2 Elicitation Study: How to indicate a window to a drone? 67
5.3 The author tested four different positioning between subject and drone 68
5.4 Experiment: Indicating a window to the drone 68
5.5 (a) Scenario Overview There are four essential elements: the user, the targets, camera, and processing unit. The user interacts with the camera in the form of pointing towards the desired target. The targets are types of visual content. For example, this includes a window that the processing unit has been trained to recognize using object detection algorithms. The processing unit is any computational system that processes the images from the camera and sends signals to the UAV. The Camera is the image feed in the moving UAV. (b) Pose Output Format: BODY-25. (c) Left-side: Points used from BODY-25 model. Right-side: Intuition on depth estimation of keypoints. When pointing backwards (transverse and sagittal planes), the projected arm on the picture will be smaller than pointing sideways (frontal plane). Using a right-angle triangle intuition, the calibration data gives us an estimation of the depth value in an arm keypoint. 70
5.6 Target Object Estimation using Pointing Gesture 72
5.7 Vector Cross Products approach to verify if two line segments are crossing each other 73
5.8 If $\vec{p}+t\vec{r}=\vec{a}+u\vec{s}$ then the two lines intersect. 74

5.9 Case when the Extended Pointing Segment PQ and Bounding Box segment AB are collinear. 75
5.10 Case when the Extended Pointing Segment PQ and Bounding Box segment AB are parallel and non-intersecting. 77
5.11 Flowchart of the 2D Approach 78
5.12 3D pointing gesture interface application flowchart. ORB-SLAM integrated with Object Detection and Pose Estimation towards pointing gesture interaction. 80
5.13 Slice of perspective projection at current user distance D_c and current building distance B_c. The 3D pointing gesture system outputs a target point T_P that, if inside a detected window area, selects that window. 85
5.14 Simulation Environment The author used Gazebo 7.0, all models used were modified from Gazebo's originals. 86
5.15 User's positioning on the experiments. The author tested the human model on eight different positions, for each Experiment. . 87
5.16 Fixed Angles Used. The author experimented with fixed angles, that meant that in some positions the human model would be pointing to the wall, or to outside the UAV's view. 87
5.17 Sample of Simulation Experiments' Results 90
5.18 Top view of real-world experiment 92
5.19 A Laser pointer was used in the experiments. As ground-truth, each user used a laser pointer to ensure they were pointing to the intended window. 92

5.20 (a) This figure shows the outputs of OpenPOSE, YOLO, and the 2D pointing gesture interface approach. This sample is a false positive; the correct window is above the selected one. This error is due to a line crossing multiple windows is ambiguous. This example is the most significant cause of error given the lack of 3D information to make a more informed decision in this approach. (b) This figure shows the outputs of OpenPOSE, YOLO, ORB-SLAM, and the 3D pointing gesture interface approach. Here, the calculation considers the estimated depth from projective transformations and monocular SLAM's point cloud, resulting in selecting the correct window. This sample is a true positive. 94
5.21 Samples from the 3D pointing gesture interface approach. (a) - (c) True positive samples. (d) - (f) False positive samples. (g) - (i) False negative samples. (j) - (l) True negative samples. 95
5.22 Samples of the 3D pointing gesture interface approach outperforming the 2D pointing gesture interface approach, given the same scene. 97
5.23 Sample of OpenPOSE's output where occlusion occurred. On both images, the head overlaid the pointing arm. On the top image, the hand's keypoint could not be "guessed." On the bottom image, even with occlusion, the hand keypoint was "guessed" close to the correct position. 99

Chapter 1

Introduction

1.1 Natural Interaction

"Natural" is a subjective term based on each individual's perception. There are many possible metrics to evaluate the interactions between humans and machines [1, 2]: performance speed, error count, time spent on a task, quality and suitability perceptions, ease of learning and others. Perhaps the ease of learning is the best metric for measuring the "natural" interaction of a system. A system that is simple to learn will feel natural [3], but different levels of technology familiarization will impact the perception of the ease of use of a system.

Direct manipulation of physical objects is natural to the common person. It is then, "natural", to try do the same with virtual objects (such as information files, documents, or actual 3D virtual objects). As research progress in multiple fields we can observe milestones that pushes human-computer interactions closer to natural interaction (NI). For example, the Mouse (in contrast to command-line interfaces) pushed us much closer to that goal. Virtual Reality goggles [4, 5], brain waves devices [6], depth cameras [7], and another multitude of sensory devices comes together in the effort to augment a machine's sensory and feedback capability, the goal seems to be the creation of artificial intuition in the machine's side, to enable true natural interaction on the human side.

Among all the devices that can be used to implement natural interactions, the camera is perhaps the most widespread. Nowadays cameras are integrated in the most accessible devices and that is why the author chose to use camera-image-based interactions in the research about natural interactions.

1.2 The Importance of the Process of Gesture Development

Natural interaction is the research on ways of interaction between humans and electronic devices, using the body as a means. There are several possibilities of interaction: through voice commands, haptic devices, olfaction, locomotion, gestures, or separate human body parts such as face, hand, thumb, eye retina. Gestures play a big role in collaboration and communication, and interest in gestural interfaces is not new, as it can be seen in [8]: an article that presents research on a method for treating gestural input so that it can be used for multimodal applications, such as interacting with virtual environments. But gestural interfaces have not become so widespread as say voice interfaces. As systems like Alexa [9, 10], Siri [11, 10] and Google Assistant [10] gain users, there are few gestural interface systems like Kinetic that have gained the popular acclamation, albeit for a little while. Why is that?

So there are a few possibilities in answering this question. First, gestures are not as incisive as words can be quick and direct. The amount of effort is also greater gesturing than talking. And also, some studies disregard user studies when choosing gestures. One example is [12], which, despite being of great importance to the area, does not seek to understand gestures in search of the most appropriate choice for the interfaces created. The lack of a Gestural Development Process can lead to using tiresome, unintuitive or non-functional gestures and can harm the performance of the application.

The process of associating a gesture to a particular action or function of the system is not trivial, because it must take into account several factors such as ergonomics, intuitiveness and objectivity [13, 14], which highlights the importance of a Gesture Development Process (GDP).

1.3 Multi-scale Gestural Interactions

The distance from the user to the computational device used to recognize the user's gesture can define a small or large-scaled interaction, in terms of distance. In the case of "selection"-like functions of a system, the user indicates a "target"

to the system. In this scenario, multi-scaled interactions also takes into account the distances between user and target.

The longer the distances between user-target and user-device, the more error accumulates on vision-based solutions. Therefore it is not trivial to develop accessible multi-scale interactions. The author developed a computer vision solution using only a regular monocular camera attached to a low-cost Unmanned Aerial Vehicle (UAV), obtaining a 0.58 F1-Score at the maximum user-target distance of 25 meters and user-device distance of 5 meters.

1.4 Statement of the Problem

- Current Gesture design techniques are not flexible enough to encompass system functions of varied complexity, or detailed enough so as to produce sufficient documentation for enabling collaboration and avoid reword on future similar system applications. Some gesture development process fail to validate their product by deploying it and testing the resulting gestural application.
- There is a lack of research on the reuse of previously defined gesture vocabularies in other cultures. The product of human interaction, gesture vocabularies, can be more valued by using multicultural perception to enable reuse and avoid rework on similar system functions.
- Gestural Interaction are usually short-scaled (up to 2 meters). The problem with large-scaled interactions is that larger distances accumulate error in the selection of a pointed target to interact with the user. Accessible Multi-Scale Interaction has the additional challenge of sensing the gestures using widespread accessible devices.

1.5 Research Questions

- How to develop a flexible process to produce gestures to system functions with varying levels of complexity?

- How to validate this solution while also creating useful gestures for common system functions?
- What are the perception differences when users from countries with different levels of technology familiarization use the same gestural interface?
- How can we combine and modify available tools to develop accessible gesture recognition interfaces that overcome error increase in a multi-scale environment?

1.6 Philosophy

This section presents the author's proposed concept of accessible multi-scale interactions. While the advance of sensory technologies provides a multitude of non-visual based devices or enhancing visual based solution for gestural interactions, in order to maximize accessibility the author has considered using only monocular cameras, as those are wide-spread and low-cost. This decision created a challenging setting for the construction of a solution that worked on a multi-scaled environment, that is when the user is up close, or very distant to the target he/she is interacting with, or capture device.

The author suggests the following steps to improve the design, evaluation and deployment of gestural interfaces for accessible multi-scale interactions:

- *Affordability and accessibility:* Chose the most widespread device to capture the interaction. The context of the intended user should be taken into account. The choice of the monocular camera as gesture detection and interaction device ensured high accessibility: monocular cameras can be found integrated in a wide range of devices and are affordable.
- *Gesture development:* All gesture development efforts should begin with a profound understanding of intended users, who are generally characterized in terms or their demographics, experience, education, leisure roles and technological familiarization. The use of a detailed Gesture Development Process will enable the following:

  - The production of artifacts during the Gesture Development Process will better document each step of the Process;
  - enabling collaboration between researchers in different places;
  - creating better representation history for each gesture elicited,
  - and finally avoiding rework on future similar gestural interfaces.
  - The cyclic structure of the Gesture Development Process makes it versatile to be used for more complex system functions, where there might be need for more interactions.

- *Multicultural Gesture Perceptions:* The participants used throughout the Gesture Development Process should belong to the system user-group as much as possible. That said, to evaluate and validate a Gesture Vocabulary on another cultural body of users will enable reuse and avoid rework on similar system functions. Additionally, the multicultural gesture perception of a given vocabulary shows promising steps towards the globalization of a given gestural interface.

- *Collaboration and Documentation by Artifacts:* The creation of artifacts to document the outcome Gesture Vocabulary on each iteration of the Gesture Development Process enables remote collaboration between researchers and facilitates the multicultural perception of such gestures. The templates depicting the Gesture Vocabulary chosen for specific system actions can be observed by future researches and used as starting points to develop gestures for similar system functions. In a society where social distance is the norm, the product of human interaction should be valued as much as possible.

- *User Positioning in vision-based solutions for UAV:* Vision based solutions utilize image feeds to analyze the environment and make decisions. In the UAV scenario this usually consists of three elements; the User that interacts with the UAV and the targets which are also objects of the user interaction. One of the challenges in this scenario is occlusion. According to the proposed results of the present work, the UAV should position itself

so that the user appears on the side of the image, rather than the center, this will drastically reduce occlusion occurrence. Meanwhile the target should also be contained on the same image.

- *Multi-scale in vision-based solutions for UAV:* Vision-based solutions utilize image feeds to analyze the environment and make decisions. Another challenge in this scenario is multi-scale interaction. Multi-scale is when the system needs to work on small (less than 2 meters) or larger distances from user-UAV, and user-targets. In this setting, the maximum distance will be determined by the limits in the recognition of the distance to the user and to the target, from the capture device. The image quality at the maximum distance should also still permit the use of specific desired technologies, such as pose estimation.

1.7 Outline of the Dissertation

The author presents the overview of this dissertation on Figure 1.1. This dissertation consists of six chapters, each of which will be briefly explained below.

- ✓ Chapter 1, Introduction, begins with the background on Natural Interaction, with Focus on Gestural Interfaces. This chapter lists the challenges, research questions, philosophy and contributions of the present work.
- ✓ Chapter 2, Literature Review, presents the related works in the process of developing gestures for multi-scaled interactions.
- ✓ Chapter 3, Gesture Development Process, presents the initial study conducted so as to develop a process that, taking into account previous work's challenges, produces intuitive and thoroughly tested gestures for functions of a system.
- ✓ Chapter 4, Multicultural Gestural Perceptions, compares and contrasts the perception of a gestural interface when used by user from two technologically different countries: Japan and Brazil.

- ✓ Chapter 5, Human-Drone Gestural Interaction, shows the continuous effort taken to produce a low-cost multi-scale pointing gesture interface.
- ✓ Chapter 6, Conclusion, is a summary of the major findings and outlines suggestions for future work.

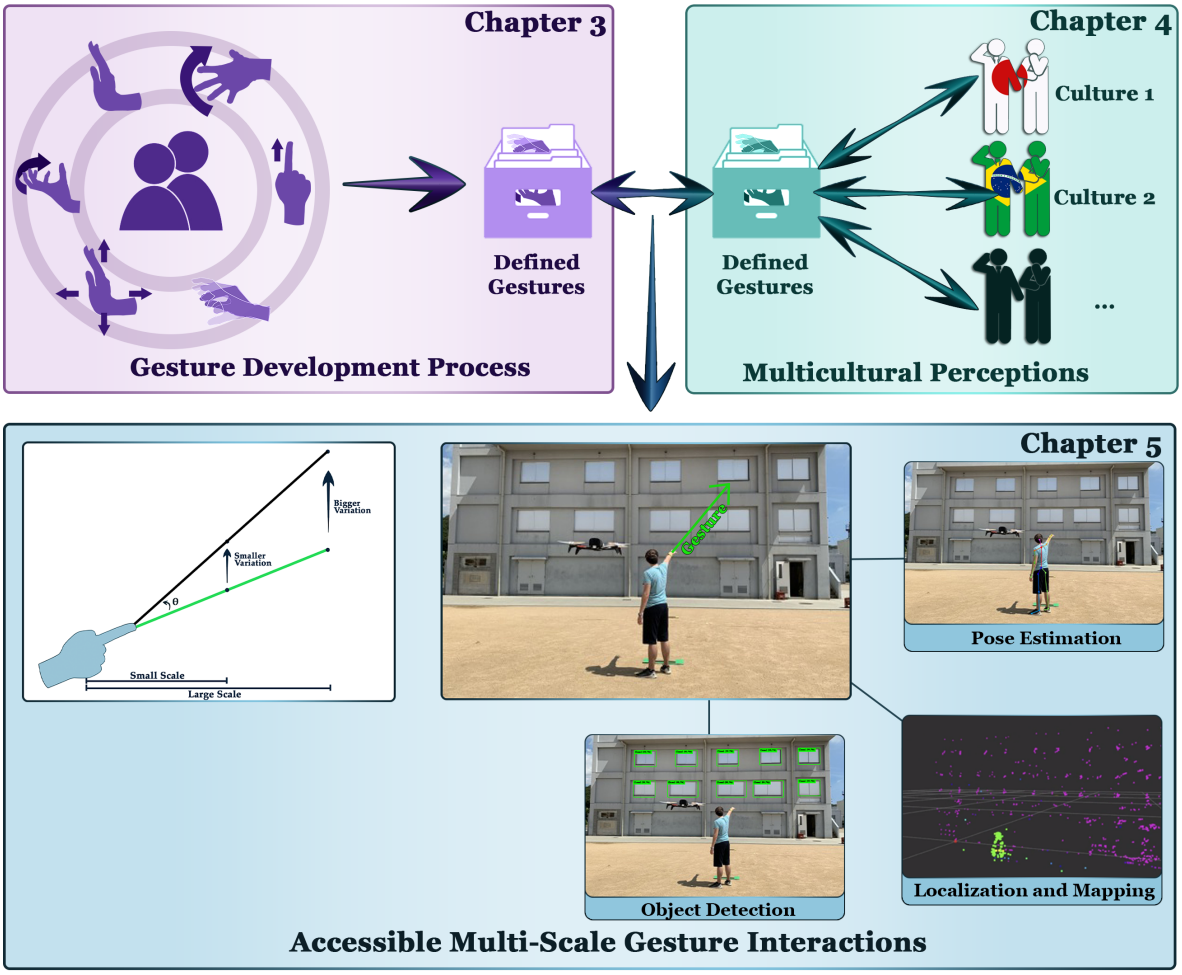


Figure 1.1. Overview of this dissertation

Chapter 2

Literature Review

This chapter presents the literature review on how previous works choose gestures for their gestural interfaces, the parameters used to compare these related works, and what is the state of the art of gestural interfaces in the context of Unmanned Aerial Vehicles (UAVs).

2.1 Gesture Development Process

The Natural Interaction area has developed rapidly in terms of scientific research and product development, so it is possible to find in the literature studies that address NI under different aspects. This fact makes the activity of listing related works more complex. Therefore, it was decided to set parameters of comparison that could guide the process of selecting these works. In this section the parameters of comparison are presented, the work considered related are detailed and, finally, a comparative analysis of them is shown.

The parameters of comparison are listed below:

- *Model of Gesture Development Process*: studies that present patterns or procedures that were used for the development of Gestural Interfaces;
- *Gesture Vocabulary*: it was considered that a work presents a Gesture Vocabulary (GV) if it presents any set, bank or group of gestures. This vocabulary must submit a description of the gesture that can be in the form of text, drawing or video;

- *Applications with Gestural Interfaces*: works that address applications, show examples (cases) that use gestures as a means of interaction. These applications can be of any type, for example, PC based or mobile devices;
- *Evaluation with Users*: studies that test somehow the gesture developed with users, thus obtaining feedback on the use of it.

The work described in [15] has two processes to develop gestures to gestural interfaces, one is focused on the technology and the other is focused on the human side. This same process is also described later in [16]. The process oriented technology is concerned with using gestures that are easily recognized by the technological devices of the time (Figure 2.1 shows an example), but it was found that users labeled gestures resulting from this process as stressful, illogical and even impossible.



Figure 2.1. Example of technology based gestures.

The process focused on the human side favors gestures easy to remember and perform by the user, intuitive, logical, functional and ergonomic. Nielsen et al. [15] divide this procedure into four stages, A, B, C and D (Figure 2.2).

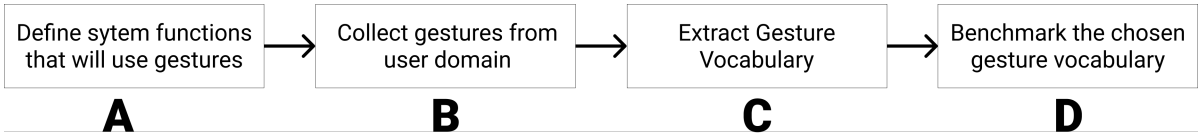


Figure 2.2. Human based process defined in Nielsen et al. [15]

In Step A, all possible functions/actions that are triggered by gestures should be set. That is, which functions of a given system will need gestures.

In Step B, the goal is to find the gestures that represent the functions found in step A. Prototypes gestures that will correspond to each function should be defined. Nielsen et al. state it should be done through interviews with volunteers,

watching them through scenarios where they communicate to an operator the same messages that they would communicate to the computer. All interviews should be recorded for later analysis. The number of people needed for this research depends on how wide the user group is and how diverse the test results are.

In Step C, the video recorded data is analysed to extract the gestures that the volunteers used in their interaction. This evaluation must take into account commonly used gestures, and how consistently the different volunteers use them. The evaluation also takes into account if the gestures are used only as static postures or if the dynamics play an important part in the interpretation of the gesture. During this evaluation, it was permitted for the person doing the analysis to define gestures different from those in the recordings.

Lastly Step D, in which the procedure evaluates the GV using three tests (Figure 2.3): "Guess the Function" Test, Memory and Stress. Each test has a score to be able to measure the quality of the final result of the GV, if not satisfied with the results, one can choose to repeat the aforementioned process by Nielsen et al..

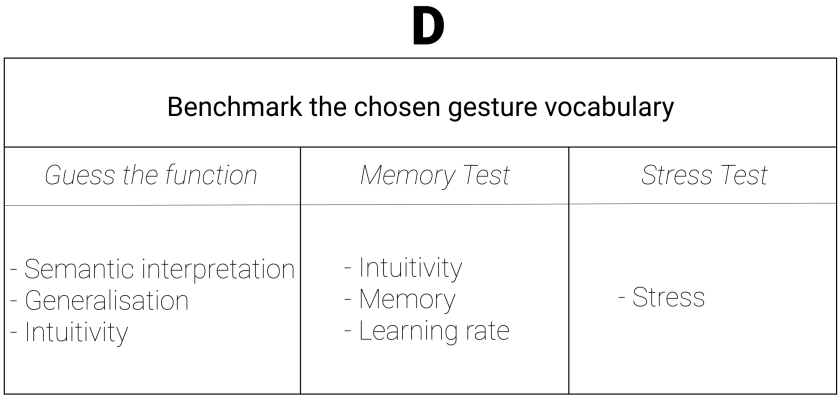


Figure 2.3. Step D evaluates the Gesture Vocabulary using three tests. Each test evaluate different attributes.

The Guess the Function Test, measures the semantic interpretation, generalisation and intuitivity of the gestures. In this test, the gestures chosen in the previous step and a list of functions are presented to the voluntaries, but the

correspondence between gesture and function are not revealed, it is left to the voluntary to intuit which functions are related to which gestures. The score is the sum of wrong guesses divided by the number of gestures.

According to Nielsen et al., memory tests measure the comfort and user familiarity with gestures, only after the user can comfortably wear the gestures is that he will remain focused on the application itself and not on the interface. First, the vocabulary of gestures is shown for each volunteer, then a slide with the functions, and 2 seconds per function. The user must then make the correct gesture to each slide if he makes the wrong gesture the slide of the procedure is restarted. The GV is reviewed between each slide restart, and the score is the number of restarts. The result is an indicator of how difficult it will be for a new user to become comfortable with the GV.

The Stress Test shows the voluntaries a sequential list of gestures, each subject should repeat the sequence a given number of times, where this amount is determined by the result of two hundred divided by the size of the GV. After completing the test, the volunteer is asked about how stressful it was every gesture, providing an overall rating for each gesture used for a long time, for example, “no problem”, “slightly tiring”, “annoying”, “painful” or “impossible.”

To test the procedure focused on the human side, Nielsen et al. use a hypothetical application that allows the user to place virtual 3D objects in a table, move them and change the style settings on them. The specific functions for this application defined in Step A were “Enable Menu”, “Select”, “Select All”, “Insert”, “Move”, “Scale”, “Rotate”, “Delete”, “Yes/Enter”, “No/undo” and “Copy-Paste.”

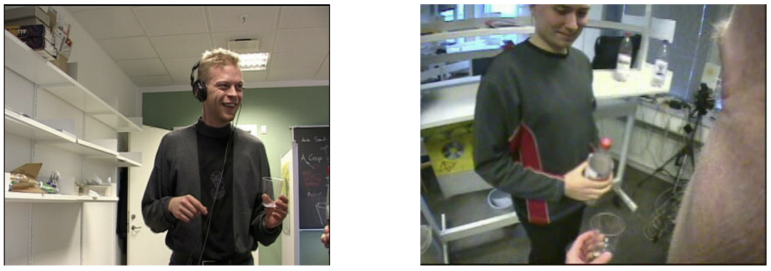


Figure 2.4. Scenario I from stationary camera showing the customer (left), and from head-mounted camera (right) showing the bartender

Nielsen et al. constructed three gesture elicitation scenarios in Step B for this application: the first is a pub where the bartender cannot listen to the customer due to excessive noise in the fictional environment, the client must then proceed with gestures to get his orders. Figure 2.4 shows this scenario.

The second scenario (Figure 2.5) consisted of a conversation about possible furniture positions, windows and doors in a room designed on paper, this scenario serves as an auxiliary of the third scenario; on the third scenario (Figure 2.5), the volunteer, now familiar with the scene, must communicate where they place objects in the scene (furniture, door and windows) using gestures. All scenarios were recorded.



Figure 2.5. Scenario II-III from stationary camera (left) and head-mounted camera (right)

In Step C, the videos from the previous stage are analyzed and the final gestures are chosen based on the aforementioned parameters, and in Step D, tests are performed with the vocabulary of gestures. The results can be seen in [15].

According to Nielsen et al., The comparative analysis of a single gesture vocabulary requires that a goal is established, which can be difficult to predefine. The best would be to compare different Gestures Vocabularies.

The research by Henze et al. [17] is based on the work of Nielsen et al. [15]. Henze et al. made some modifications to the gesture development procedure described by Nielsen et al. and applies it to develop the concept of a software for playing music, which was not deployed in the study.

The process (Figure 2.6) described by Henze et al. also consists of four steps: Context of Use and Functions, Participatory Design, Definition of Set of Gestures and the last stage called Evaluation and Improvements.

In the Context of Use and Functions Step, Henze et al. sought to understand

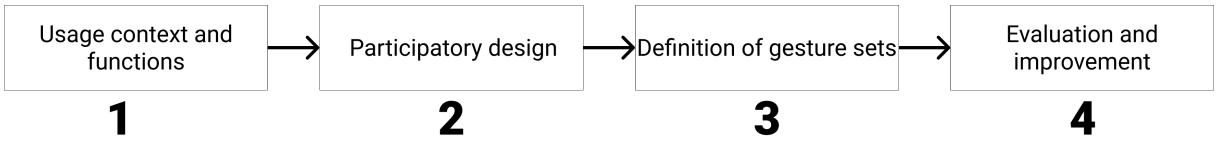


Figure 2.6. Henze et al. [17] gesture development process was based on Nielsen et al. [15]

the context of the use of applications that play music to define the functions that their hypothetical music player should have. This study was divided into two parts: in the first part the focus was to analyze the need and purpose of listening to music in different situations. This was done through interviews with volunteers; in the second part a prototype that recognizes covers CD's via webcam and plays a song on an album was presented to volunteers, in order to gain initial feedback on the functionality and the desired behaviour of the hypothetical system. At the end of this stage, it is obtained an initial set of system functions for the hypothetical application. Nine volunteers participated in this stage.

The objective of the participatory design step is to validate the set of functions defined in the step 1 and to derive multiple gestures, for each function, through participatory design techniques. This study was also divided into two parts. The first part validated the previous set of functions and more information were collected about the habits of listening to music from willing participants through personal interviews. The final set of functions were, "play", "stop", "pause","next", "previous" and "change volume".



Figure 2.7. Exemplary camera image from the participatory design study by Henze et al. [17]

In the second part the focus was to collect potential gestures. For this, the volunteers performed gestures for each function present in a list of functions, given in front of a computer equipped with a camera (Figure 2.7). Participants should announce orally what function they wanted to perform before gesturing. The system functionality was simulated by an operator, according to the respective function announced by the participant volunteer. At the end of this stage, recordings were obtained with candidate gestures and they were classified as: static, dynamic or hybrid. Figure 2.8 shows an example of each type of gesture. Ten new volunteers participated in this stage.

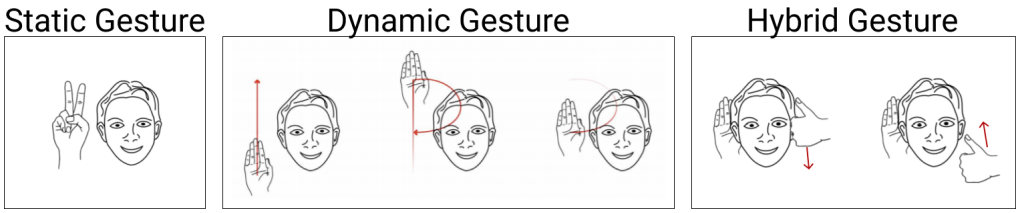


Figure 2.8. A static gesture can be defined as a specific position of the hands or body. A dynamic gesture changes the position over time to complete the gesture. A hybrid gesture might be static on one hand and dynamic in the other.

In the Step of Defining the Set of Gestures, the goal is to formalize and define the gestures. Using recordings of previous steps, it was possible to analyze the gestures chosen taking into account concepts also considered in Step C of Nielsen et al., and popularity or frequency of the candidate gestures. At the end of this step, different from step B of Nielsen et al. that defines a gesture prototype for each function, the work of Henze et al. defines two prototypes gestures, a static (Figure 2.9) and other dynamic (Figure 2.10) for each role to be analyzed in the next steps.

In the Evaluation and Improvements Step, both sets of prototypes gestures are evaluated and eventually refined. In addition, other research about the functions is made, about if there are other functions that would be essential to control a music player. The Evaluation and Improvements Step is divided into three parts. The first part assesses the already determined set of functions and the opportunity for the participant comment on new functions. It was used a

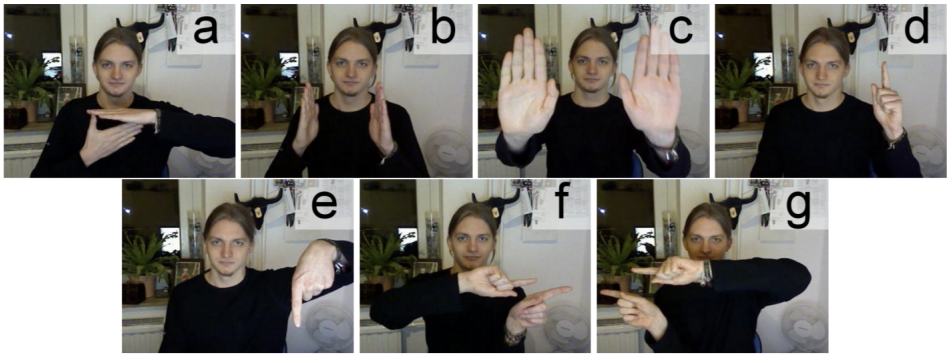


Figure 2.9. The set of static gestures (from a to g: Start, pause, stop, increase volume, decrease volume, next, and previous)



Figure 2.10. The set of dynamic gestures (from left to right: Start/pause, stop, increase volume, and next)

comparative scale of 1 (unnecessary) to 7 (essential) to evaluate the set of functions and all functions were classified with a mean and median of at least 5.5. The second part consists of a user test using an environment similar to the one described in Participatory Design Step, in which each participating volunteer should perform static and dynamic gestures of each function displayed by a list of functions. After the evaluation was over, each participant classified simplicity, intuitiveness, consistency, ease of use and ease of memory of gestures. The scale ranged from 1 (not a bit) to 7 (perfect) and, in the latter part, the participant could give their opinion on some function that he felt missing. Two people did not mention extra features, others cited functions such as “go to random music” and “edit playlist”, giving examples of gestures to them. Twelve volunteers participated in this stage. The final stage of the work of Henze et al. does not use

any of the tests described in Step D of Nielsen et al..

The work of Epps et al. [18] develops gestures to functions of horizontal human-computer systems, such as tabletops, to investigate gestural preferences of users of these systems. The authors defined 36 functions, the main ones being: “select”, “open”, “move”, “scroll”, “design”, “cut / paste”, “rotate” and “zoom”. They were made by volunteers on a horizontal human-computer system. For each function, the screen displayed a static image and asked the volunteers to use gestures in the hypothetical performance of functions, using the objects shown in the picture. Twenty volunteers participated in the study. The frequency was used as gesture selection criteria.

The work of Choi et al. [19] focuses on showing that the most popular gestures on a user survey are not necessarily the best choices. To validate their hypotheses, they use as a case study a smart-home system basing their study on the analysis of the works: [15, 17, 18], among others. These proposals have development processes focused on gestures in which frequency is used as a parameter of choice. Each candidate thinks in a restricted universe of gestures possibilities for each function, if one of the candidates is an interfaces designer or even an experienced user, then his suggestions can be valuable, so it is likely that significant gestures are neglected because of their low frequency.

Choi et al. created hypotheses that characterized failures previously observed, then conducted two experiments to identify the described shortcomings. The first experiment focused on mapping gestures for interaction with structures of a house and household, and studied the possible reasons for the choices of each participant.

In the second experiment, participants were asked to choose the most appropriate gesture to each function from a list that included most of those from the first experiment. The functions were defined to the scenario of a smart home. In the first experiment, there were 30 participants, in the second 28. All participants in the second experiment were present in the first. After both experiments, each participant assessed the suitability of the chosen gesture.

During a third experiment the results of both experiments were analyzed and compared, it was then found that most of the participants changed their gesture

of choice in the second experiment, also only some of the gestures performed in the first experiment were selected as the top during the second experiment, thus showing that gestures with few occurrences may be ideal.

Choi et al. did not conclude what would be the ideal gesture to the actions chosen within the setting of a smart home, claiming that the focus of the work was not it.

Jeong, et al. [20] propose a specific control system for TV which has single camera. This system exploits an alternative to remote control in the most frequently used functions to watch TV. To determine these functions, Jeong, et al. used an open questionnaire to the public, thus obtained the following functions as most popular: volume, channel, stop / play and fast search. There was no specific process for defining gestures corresponding to the mentioned functions, those were determined by the developers themselves. The template of the chosen gestures was visible on the TV screen.



Figure 2.11. Demo system for the human to TV interaction

After the development of the prototype work, 15 volunteers used it and answered a questionnaire in which most participants agreed that the proposed system can be used in a convenient and fun way, after getting acquainted with him, but there were complaints about the comfort and diversity of the gestures, which highlights the importance of a good gestures development process to improve the user experience.

The work of Wu et al. [21] develops a set of design guidelines for gestures, consisting in a 3-stage participatory design method used to generate a set of gestures for interacting with in-vehicle information systems. Wu et al. did not implement the resulting gestures to an actual system.

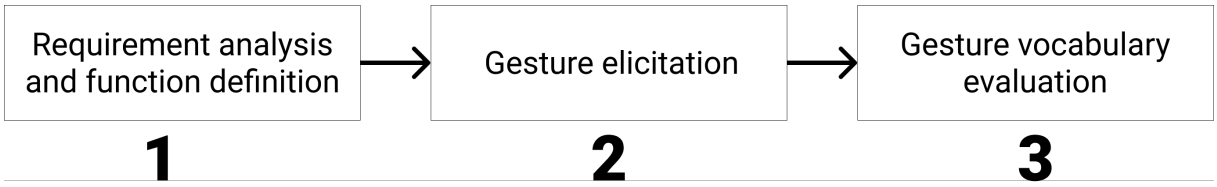


Figure 2.12. Wu et al. [21] design guidelines steps

Figure 2.12 shows the steps used on the design guidelines, they show steps very similar to Nielsen et al. [15] and Henze et al. [17], starting by defining the functions of the system that will later receive triggering gestures, following by an video-recorded elicitation study with volunteers that generated two gestures for each function, similar to steps 2 and 3 of Henze et. al.. The last step of Wu et al. gesture development was about participant consensus regarding gesture choices for a given system function. Here, instead of designing gestures, participants were required to choose two gestures for each target task verbally, similar to the "guess the function" test by Nielsen et al.

2.1.1 Comparative Analysis

In Table 2.1 there is a comparison between the works cited in this chapter and gesture development process by the author. It is possible to observed that there are works on the development process of gestures that do not validate their final vocabulary in the evaluation of an actual gestural interface application. Other works only develop a Gesture Vocabulary without worrying about its development process. Some studies do not outline a template of their final gestures, which can hinder understanding or even the repetition of an experiment conducted on them. Also, there are works that are not concerned with assessing the end-user experience. The aforementioned are the issues the author tried to address in the proposed present work.

Table 2.1

Comparison of related works on gesture development process

Work	Dev. Process	Vocabulary	Templates	Applications	User Evaluation
Nielsen et al. [15]	✓	✓	-	-	-
Henze et al. [17]	✓	✓	-	-	-
Epps et al. [18]	✓	✓	-	-	-
Choi et al. [19]	✓	✓	✓	-	-
Jeong et al. [20]	-	✓	✓	✓	✓
Wu et al. [21]	✓	✓	✓	-	-
Ch. 3 Work	✓	✓	✓	✓	✓

2.2 Multicultural Perceptions

Culture is the beliefs, value systems, norms, myths, and structural elements of a given organisation, tribe, or society [22]. In a society each day more globalized it is valid to reseach on how interfaces will impact user perceptions on various aspects of different cultures.

The work of [23] investigates the comprehension and usability variances in web usage in the multicultural south African society. At the time, most of the web interfaces were in English, and it corresponded to the second or even third language of the participants on the study. A total of 246 subjects took part in the study. They were divided according to they primary, secondary and ternary languages. The result showed that there was indeed a perceived difference in the understanding of the user interface whether the user was interacting or not with their primary language, but this study failed to take into consideration the different technological levels of the participants involved. The authors even stated that the different technological levels among the participant groups could have had an influence on the results, but they didn't take that into account.

2.3 Multi-Scale Gesture Interface and UAVs

Gesture recognition [24] has been researched using various devices such as RGB cameras, depth cameras [25, 26], radar-based sensors [27], capacitive sensors [28], sensorized gloves [29], electromyography sensors [30, 31, 32, 33], Wi-Fi [34] and others. Each device has its own limitations, and in this sec-

tion the author will summarise the use of these devices on detecting pointing gestures and/or robot control.

The work of Tolgyessy [35] investigates the use of pointing gestures to give target locations to Roomba-like robots. It uses an RGBD sensor (Kinect) mounted on a 2-DoF robot to interpret the pointing gesture from a person standing 2-3 meters from the setup. They used third-party software to get the body points from the user. In order to choose which data to use to determine the pointing gesture, they investigated combinations of wrist-elbow, wrist-shoulder, and wrist-head. Wrist-elbow gave a better result in terms of accuracy. The limitations were that the user was always facing the robot, distance was limited (up to 3 meters), and the user had to hold the pointing gesture with one hand while the other hand performed another gesture that enabled the robot to act on the pointing gesture. The user also had to point to a place inside the Field-of-View (FoV) of the robot, which limited the pointed places to very short distances in the indoor environment. The system was not suitable for outdoor use because of Kinect's hardware limitations.

The work of DelPreto [31] presents a gesture-based interface for human-robot interaction using wearable sensors. "A real-time clustering algorithm processes streaming data while only maintaining a key subset of observations. This building block is used to adaptively threshold muscle and motion signals, detecting arm stiffening and fist clenching from EMG signals and rotation gestures from IMU signals. These predictions are fused with IMU processing to interpret left, right, up, or down gestures." In this work, participants used the interface for real-time robot control by remotely piloting a UAV around obstacles. Some of the sensors used here required the participants to be connected through cables to a Data Acquisition (DAQ) device, thus restricting the applicability of this means outside of a controlled environment. This means of interaction also won't provide enough input to infer location to a robot; therefore, it cannot be used to detect pointing gestures.

The work of Liu [36] presents a method for automatically detecting the teacher's pointing gesture on the electronic whiteboard. They detect pointing gestures inside an indoor short-ranged environment, like a classroom. They use

third-party software to identify body poses, and image form filtering to identify the whiteboard. If the hand point is inside the whiteboard, the system identifies the teacher pointing out the whiteboard. This system is limited to an indoor environment where the location of the individual is very near the whiteboard (target) so that they overlap.

The work of Gromov [37] uses a pointing gesture to control UAV movement until a desired location. The experiment takes place indoors on a small environment. This system uses an IMU sensor on one arm and the other arm holds a button to control the UAV's movements. It is not possible to point directly to the desired location. The change in the user's arm position indicates change in the UAV's positioning.

Azari et al. [38] detect pointing gestures in individual RGB-D frames. A CNN-based detector is used to find hands and faces in the images, then the corresponding depth channel pixels is examined to obtain full 3D pointing vectors. It can be noticed how many works rely on sensors other than a monocular camera (RGB) to detect gestures and guide drones on tasks. But drones usually are already equipped with RGB cameras, adding extra sensors like depth cameras or scanning lasers range-finders might be impossible due to weight limits on UAVs, especially on lightweight drones, or even financial constraints.

The work of Mirri et al. [39] shows a recent review (2019) about Human-Drone Interaction. It categorizes works based on the following dimensions: The drone role (e.g., sensing tool), the context of use (e.g., helping in rescue missions), indoor VS outdoor usage, the interaction mode (e.g., gestures). The present work would fit the dimensions as exemplified before (sensing tool, help in rescue missions, using gestures). Mirri et al. [39] also mention that scenarios involving computer vision solutions for UAVs used in collaboration with the emergency workers, such as firefighters, opens challenges in the way the operator can interact with the UAVs considering that he/she is not fully dedicated to the drones, but involved in search and rescue tasks. Then, the way to develop a Pointing Gesture Interface that diminishes the effort required by the operator, and at the same time allows the drone to support the mission effectively, is one of the utmost importance.

There are multiple studies that research the use of gestures to navigate a UAV [40, 35, 31, 37]. However, the author investigates the collaboration between UAVs and humans to relieve human task load and reduce UAV's error by processing periodical human-input. This collaboration takes shape on an outdoor environment, where long distances make the use of depth cameras difficult due to hardware limitations. Moreover, lidars like Hokuyo [41] cannot gather enough information on the pointing gesture and building's windows simultaneously. Furthermore, it is not possible to use wearable devices like electromyography sensors to determine where a user is pointing, as this could also impair dexterity or task performance in a firefighting environment. Therefore, the present work focuses on using a monocular camera to produce a system that enable correct window selection decisions in indoor and outdoor environments without using external devices, even on low-cost UAVs. Table 2.2 shows a comparison between the present and related work.

Table 2.2

Comparison of related works on gesture interface for UAVs. Short-scale are for distances less than 2 meters. The author tested large-scale until 25 meters.

Work	***Gesture***	***GDP***	***Indoor***	***Outdoor***	***Wearable***	***Depth Cam***	***Monocular Cam***	***Short-scale***	***Large-Scale***
Nandakumar et al. [26]	✓	-	✓	-	-	✓	-	✓	-
Tolgyessy et al. [35]	✓	-	✓	-	-	✓	-	✓	-
DelPreto et al. [31]	✓	-	✓	-	✓	-	-	✓	-
Liu et al. [36]	✓	-	✓	-	-	-	✓	✓	-
Gromov et al. [37]	✓	-	✓	-	✓	-	-	✓	-
Ch. 5 Work	✓	✓	✓	✓	-	-	✓	-	✓

Chapter 3

Gesture Development Process

This chapter shows the construction of a Gesture Development Process that can assist in deciding which gesture would better represent a certain function (action) of a system. This GDP produces artifacts during its execution. The final product of this Process is a Gesture Template depicting the finalist and best gestures for the given functions.

3.1 Overview

The GDP here described has three steps as illustrated in Figure 3.1. Each step is detailed in sub-steps and includes the elaboration of artifacts. The first step involves determining the functions. The second step focuses on identifying test scenarios, analysis and recording of recordings and culminates in the definition of the vocabulary of gestures. The last and third stage consists in refining the vocabulary of gestures until its final version. In the following sections each step is detailed.

3.1.1 Step 1: Define Functions

In the first step, Define Functions, the problem domain is analyzed and the functions of the application to be developed are determined. This stage is not concerned with any kind of gesture, only with determining the functions to which you want to assign gestures.

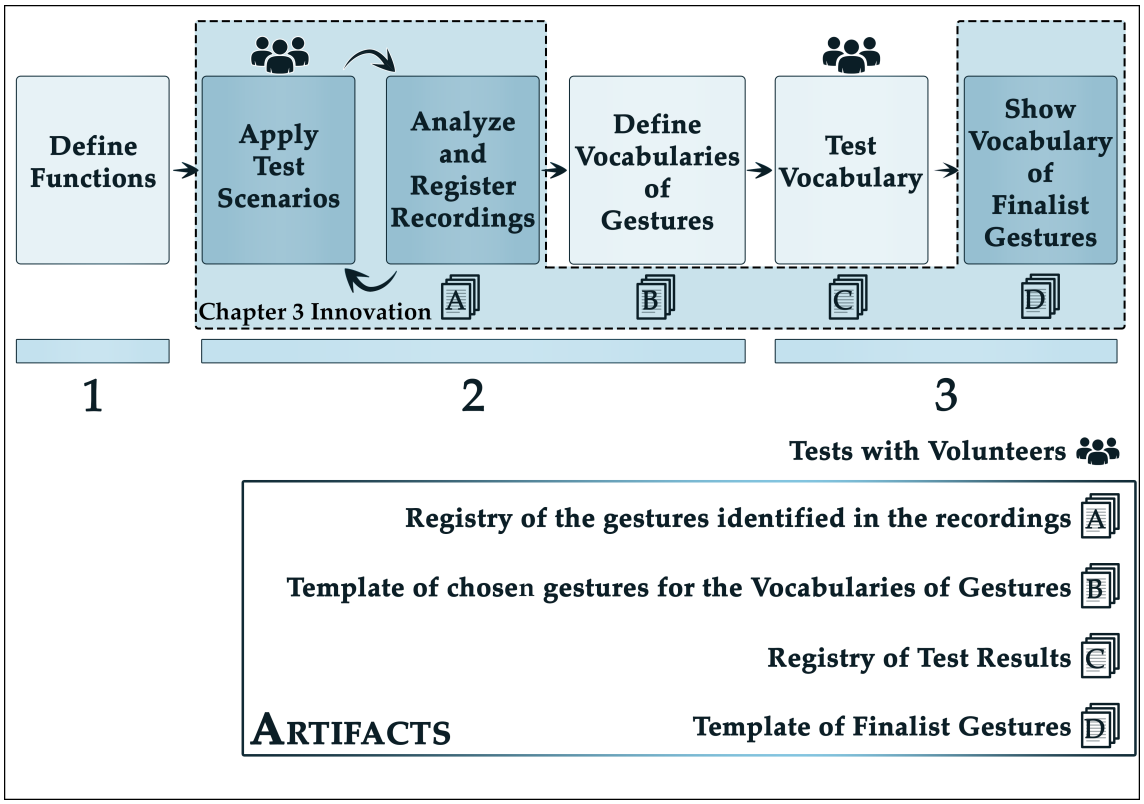


Figure 3.1. Proposed Gesture Development Process

3.1.2 Step 2: Generate Gestures

The second step consists of three sub-steps as shown in Figure 3.2. The first sub-step, Apply Test Scenarios, requires volunteers in order to perform tests to generate prototype-gestures for the functions determined in the previous stage. One or more test scenarios, which abstract technical thinking and at the same time contextualize the functions to stimulate the volunteers to execute them, are created. All the interactions at this stage are recorded. The number of participants should be directly proportional to the diversity of the gestures obtained for each function: in other words, as the tests are applied, the more diverse are the gestures for each function, the more volunteers are necessary.

During the second sub-step, Analyze and Register Recordings, an artifact is created containing the analysis and registers of the identified gestures in the recordings. In this sub-step a decision must be taken: if the results are not considered sufficiently favorable, it is necessary to go back to the previous step,

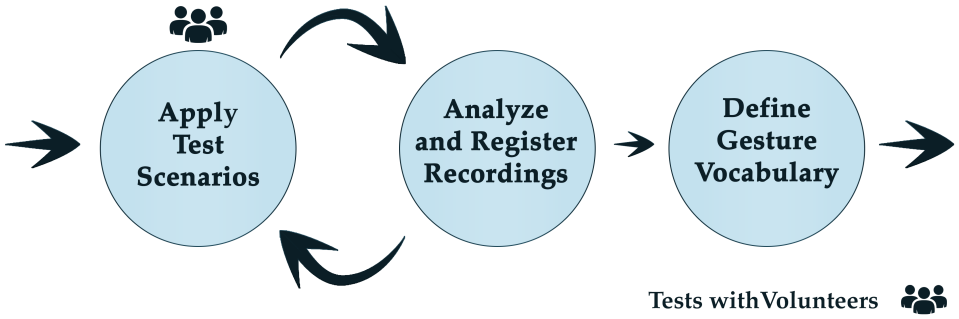


Figure 3.2. Generating gestures from elicitation studies in a cyclic manner

but now with prototypes-gestures as final suggestions for volunteers, in order to refine the gestures development process. This cyclic structure is an innovation compared to the previous related work and allows the application of the gesture development process in more complex system functions. Otherwise, go forward to the third sub-step.

In the third sub-step, Define Vocabularies of Gestures, the most favorable gestures are determined, based on the previously produced artifact, taking into account the most frequent, the ones chosen by the designer and those whose movement and posture do not compromise physical structures and articulations. A template of the chosen gestures is then generated to be used in the next stages.

It is worth mentioning that one can determine various gesture-vocabularies. At the end of this second step two artifacts should have been produced: the registers of the identified gestures in the recordings and the template of the chosen gestures.

3.1.3 Third Step: Evaluation

The third and last step has two sub-steps. In Test Vocabulary the process evaluates the N gesture vocabularies through three tests: Attribution of Semantics, Memory and Stress Tests, as described in Nielsen, et al. [15]. Each test has a score and at the end it will be possible to decide the best vocabulary through the score comparison. In the case of only one vocabulary the score it will be used to infer its quality.

In the following tests, the N vocabularies obtained are tested separately, the

same volunteers evaluate all vocabularies, it is important to vary the vocabulary which each volunteer will start evaluating, in other words, in the case of being two vocabularies, a volunteer will first evaluate the vocabulary A then B, and other volunteer would start with B and later evaluate A, trying to make the numbers of volunteers starting with A equal to the ones starting with B.

The Attribution of Semantics Test consists in presenting to the volunteer a template of the GV and a list of functions, however it's not revealed the correspondence between the gestures and functions. It is then asked for the volunteer indicate which gesture corresponds to each function. The score is the sum of wrong guesses, divided by the number of gestures.

The Memory Test measures the familiarity of the user with the gesture. Only after the user is acquainted with the gestures is that he will able to maintain the focus in the task at hand rather than to how operate the interface. First it is shown a GV to each volunteer, later, a slideshow with all the functions, staying 2 seconds in each function. The volunteer is then asked to do the correspondent gesture of that function, until being able to get all functions right, when mistaken the presentation is restarted and the vocabulary is revised. The score of the memory test is calculated by the number of restarts necessary, resulting in a measurement of the difficulty for a new user to become familiar with the gesture vocabulary.

The Stress Test shows the volunteers a sequential list of gestures. Each volunteer should repeat the sequence a determined number of times, enough so that the he can infer any possible discomfort. At the end of the test, each volunteer is questioned about how stressing is each gesture, giving a general classification for them: "no problem", "slightly tiresome", "annoying", "painful" or "impossible".

At the end of all tests, in Show Vocabulary of Finalist Gestures, the vocabularies can be compared through their scores and classifications, then it would be drawn a final template for the gestures of the chosen vocabulary, so as to aid the implementation of these gestures to their respective functions on the application.

3.1.4 Artifacts

Some steps in the proposed GDP (Figure 3.1) produce artifacts. The first artifact to be produced is "Records of Gestures Identified in Recordings" and, at the end of this Gesture Development Process, there will be one or more versions of this artifact, since each time the "Analyze and Record Recordings" activity is executed, a new version of this artifact is created. The second artifact is "Template of Chosen Gestures for Gesture Vocabulary (s)", resulting from the activity "Define Gesture Vocabularies". A template must be generated for each chosen gesture vocabulary, remembering that the developer can choose to determine one or more vocabularies, based on the results obtained with the "Record (s) of the Gestures Identified in the Recordings". In "Test Vocabulary (s)", a "Test Results Report" is generated and in this data are found from one or more vocabularies in the Semantics, Memory and Stress Tests. From this artifact, it is possible to choose which vocabulary obtained the most favorable result, to become the final vocabulary, or in the case of a single vocabulary, to infer its quality. In "Display Final Gestures Vocabulary" a "Template for Final Gestures" is generated, in which the developer can find representations of the gestures of the chosen final vocabulary, and then use this artifact as a basis to implement the gestures in the application.

These artifacts are products of the human interaction required to elicit gestures for the system functions, these artifacts should be properly represented in the form of videos, images or at least text descriptions. It is important to depict each gesture in relation to which system function in is representing. These artifacts can enable remote collaboration and avoid rework in many cases. They serve as a means to extract maximum value from human interaction, specially in a society where social distance is the norm. The author hopes that these artifacts can in the future create a dataset that relates gesture and system functions.

3.2 Validating the Gesture Development Process in the Three-dimensional Navigation Scenario

The application of the GDP described above was demonstrated in the development of a new program that would feature natural forms of interacting with a cube. The author's idea was to develop gestures for common functions in 3D environments. A cube was defined as the object to be manipulated and the name chosen for the application was CuboNI (Natural Interaction). Figure 3.3 shows the summary of the GDP on the 3D Navigation scenario. Observe that certain activities can be performed multiple times. The number of volunteers can also vary according to the needs of the step. The execution of GDP for the creation of gestures for CuboNI is explained in the following sections.

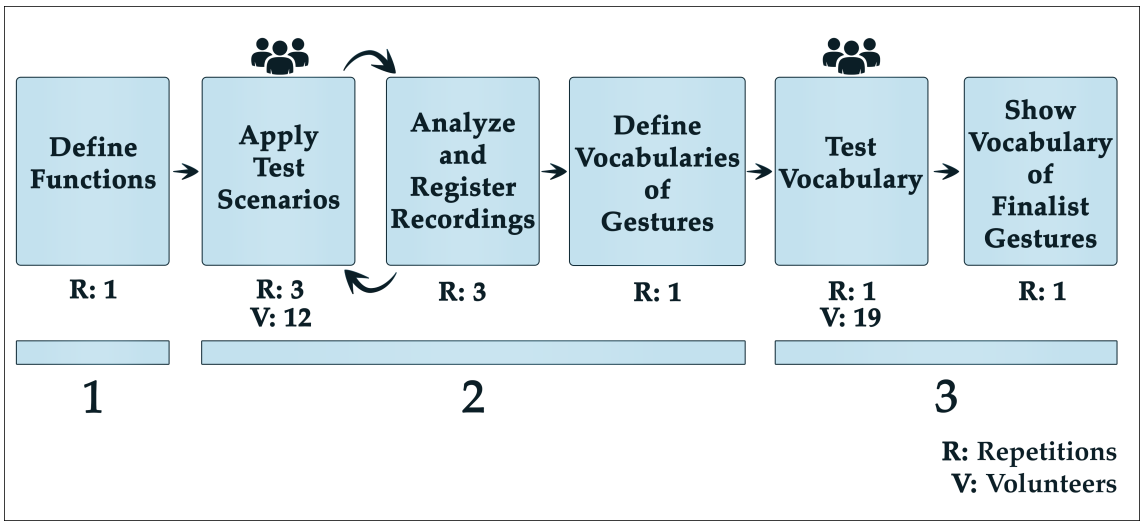


Figure 3.3. Proposed Gesture Development Process on the 3D Navigation Scenario

3.2.1 First Step

The first step is to select which function will receive gestures as triggers. The chosen functions were "Rotate", "Scale" and "Translate", as these are the main two-dimensional and three-dimensional geometric transformations [42], and "Select" and "Stop", because they are related to the beginning and end of interactions in general.

3.2.2 Second Step

In the “Apply Test Scenarios” step of the second step, a case scenario application was created to stimulate the volunteers in the production of gestures for the functions mentioned before. The CubeMK (Mouse and Keyboard Cube) featured a colorful cube (Figure 3.4) that could be selected, rotated, translated, scaled and stopped (when stopped, no interactions could be performed with the cube), all through the mouse and keyboard input. The idea was that the volunteers could observe this interactions, then they were asked to perform gestures that, in their minds, would seem intuitive to trigger the before functions. This test was first carried out with 12 volunteers, all students from Computer Science bachelor in Brazil, all tests were recorded.

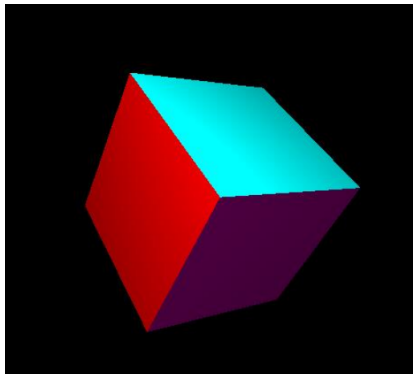


Figure 3.4. CubeMK

With the recordings from the first round of tests, it was time to proceed to the second step, “Analyze and Register Recordings”. Here the recordings were analyzed and the gestures captured were registered as shown in Figures 3.5, 3.6, 3.7, 3.8 and 3.9. The most popular gestures were S1, R3, T2, E1 and P2. The decision here was made in favor of another round of tests, in order to see if there would be many more new gestures or if 12 volunteers were enough.

The first step was then repeated with other 12 volunteers, only now the tests would provide the gestures in Figures 3.5, 3.6, 3.7, 3.8 and 3.9 as options at the end of each test, so that the volunteer could change his choice if he preferred one of the gestures presented.

On the recordings of the second round of tests, the registry showed one new



Figure 3.5. Gestures generated for the "Select" function.

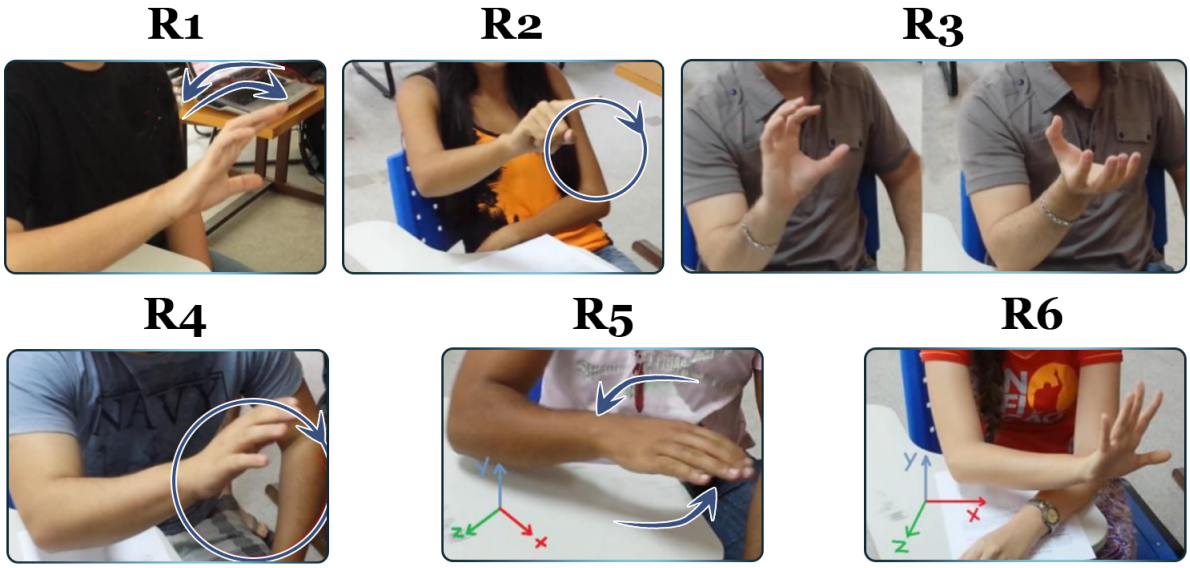


Figure 3.6. Gestures generated for the "Rotate" function.

gesture for "Select", two new gestures for "Rotate" and "Translate", and three new gestures for "Scale" and "Stop". The most popular gestures now were still S1, R3, T2, E1 and P2. This was a good indicator that the first 12 volunteers produced a good variety of gestures, yet it was decided to do a last round of tests with students from different areas, to see if the results would be alike.

Back in to the first step other 12 volunteers repeated the last test, but now with more gesture options at the end of each test. The volunteers from this third round of tests were students from several majors including: Electrical Engineering, Law, Public Administration and Medicine. Going forward to step 2, the registry showed two new gestures for "Select", "Rotate" and "Scale", four

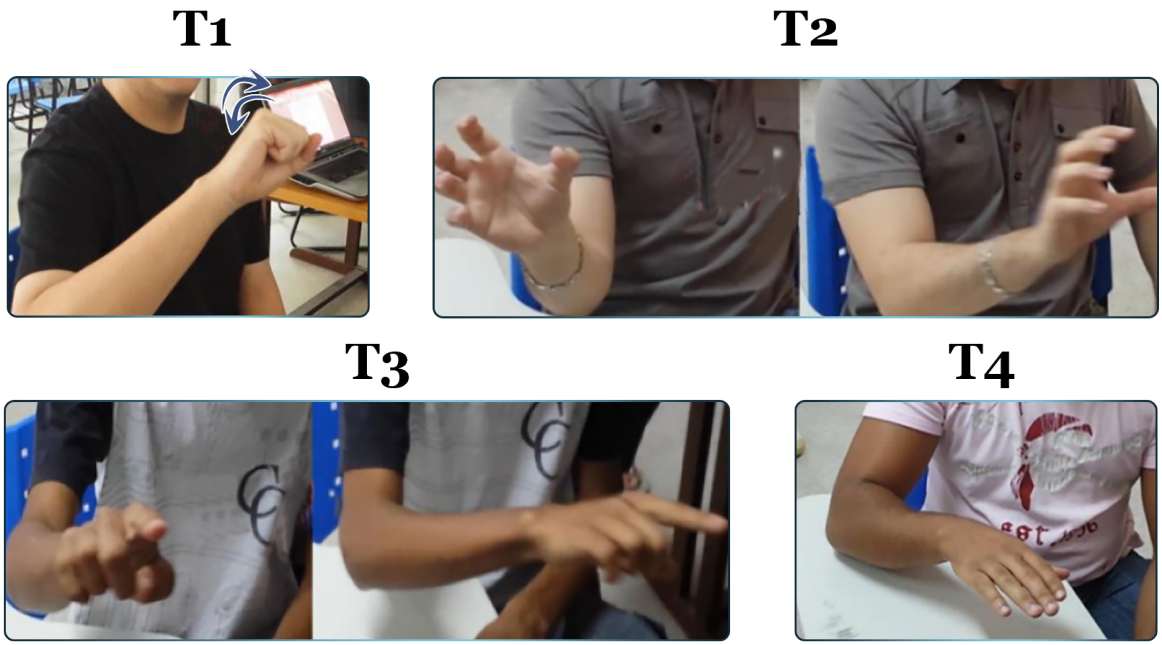


Figure 3.7. Gestures generated for the "Translate" function.

new gestures for "Translate", and no new gestures for "Stop". The most popular gestures with this last group of volunteers were S1, R3, T2, E4 and P2. But in the total 36 volunteers the most popular gestures were still S1, R3, T2, E1 and P2. Seeing as that from 12 to 36 volunteers only one gesture changed (E1 to E4) the results were satisfactory and it was decided to proceed to the next step "Define Vocabularies of Gestures".

It was selected two gestures vocabularies; one containing: S1, R3, T2, E1 and P2, the most popular choice between the 36 volunteers, and another containing: S3, R7 (Figure 3.10), T1, E4 and P1, less popular choices but considered good to be compared to the first vocabulary, in the next stage. The template produced in this third step can be seen in Figures 3.11 and 3.12

3.2.3 Third Stage

In the third stage, Attribution of Semantics, Memory and Stress tests were performed with Gesture Vocabularies 1 (Figure 3.11) and 2 (Figure 3.12). This stage counted with 19 volunteers, students from several majors: Civil Engineer-

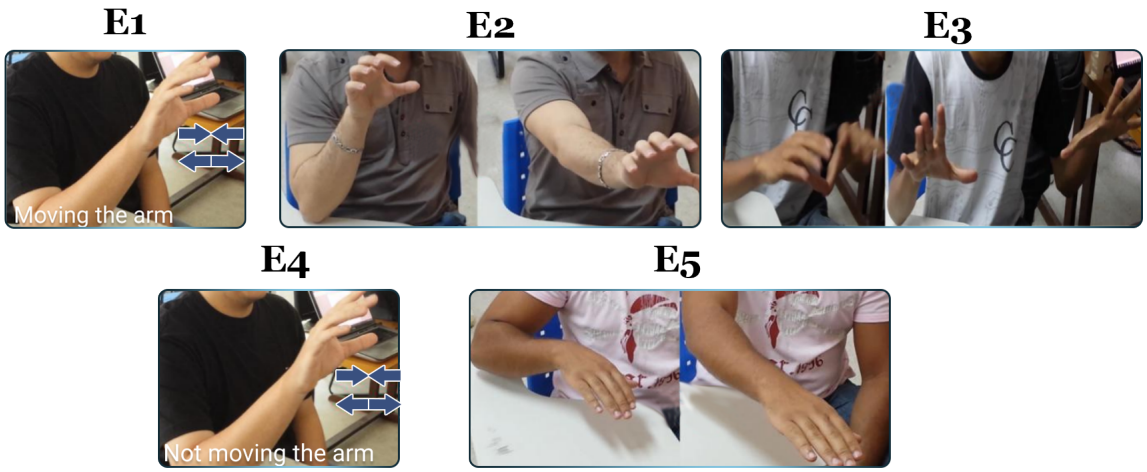


Figure 3.8. Gestures generated for the "Scale" function.

ing, Medicine, Computer Science, Electric Engineering, Law, among others.

In the Attribution of Semantics test the templates (Figures 3.11 and 3.12) and a list with the functions before commented were presented to each volunteer. It was then asked for the volunteers to indicate which gesture corresponds to each function. The results for each Gesture Vocabulary can be seen in Table 3.1. Bear in mind that the score is the sum of wrong guesses divided by the number of gestures, so a perfect score would be zero.

Table 3.1

Attribution of Semantics Test Results

	Gesture Vocabulary 1	Gesture Vocabulary 2
Average	1.4	1.8
	Among 19 participants, there was a total of 7 wrong guesses of the 5 system functions	Among 19 participants, there was a total of 9 wrong guesses of the 5 system functions

The Memory test showed to each volunteer the GVs 1 and 2 and a slideshow with one slide for the name of each function ("Select", "Rotate", "Translate", "Scale" and "Stop"). The slideshow would linger 2 seconds on each slide. When reading the name of the function in a slide, the volunteers were asked to do the correspondent gesture of that function, if they did a wrong gesture the presentation was restarted.

The score of the memory test was calculated by the number of restarts nec-



Figure 3.9. Gestures generated for the "Stop" function.

essary until the volunteer got all gestures correct. This was carried out twice for each volunteer since there were two GVs. The results can be seen in Table 3.2. The best result would be zero.

Table 3.2

Memory Test Results

	Gesture Vocabulary 1	Gesture Vocabulary 2
Average	1	2
	Among 19 participants, there was a total of 1 restart	Among 19 participants, there was a total of 92 restarts

The Stress Test showed the volunteers a sequential list of the gestures from each GV. Each volunteer was then asked to repeat the sequence 50 times, for each gesture in each GV. At the end of the test, each volunteer was questioned about how stressing was each gesture, giving a general classification for each of the gestures in both GVs: "no problem", "slightly tiresome", "annoying", "painful" or "impossible". The result can be seen in Tables 3.3 and 3.4.

In the end the Gesture Vocabulary 1 was slightly better them the GV 2, but the gesture to "Scale" of the GV 2 achieved significantly better results to that of GV 1, and so on "Presenting Final Gesture Vocabulary" the final vocabulary

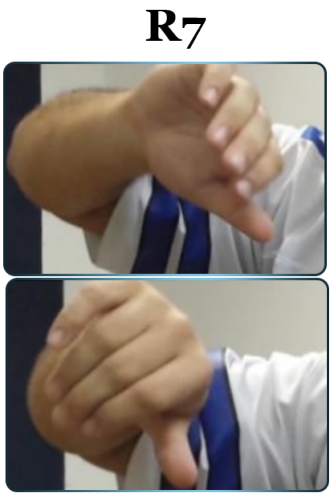


Figure 3.10. Gesture R7

Table 3.3

Stress Test Results on Gesture Vocabulary 1

	Select	Rotate	Translate	Scale	Stop
"No problem"	19	15	19	10	19
"slightly tiresome"		4		8	
"annoying"				1	
"painful"					
"impossible"					

Table 3.4

Stress Test Results on Gesture Vocabulary 2

	Select	Rotate	Translate	Scale	Stop
"No problem"	15	10	18	16	19
"slightly tiresome"	4	5	1	3	
"annoying"		4			
"painful"					
"impossible"					

of gestures chosen contained all the elements from GV 1, but "Scale" which was taken from GV2. The gesture template with the final GV (S1, R3, T2, E4 and P2) can be seen in Figure 3.13 , as well as the artifact "Template of Final Gestures".

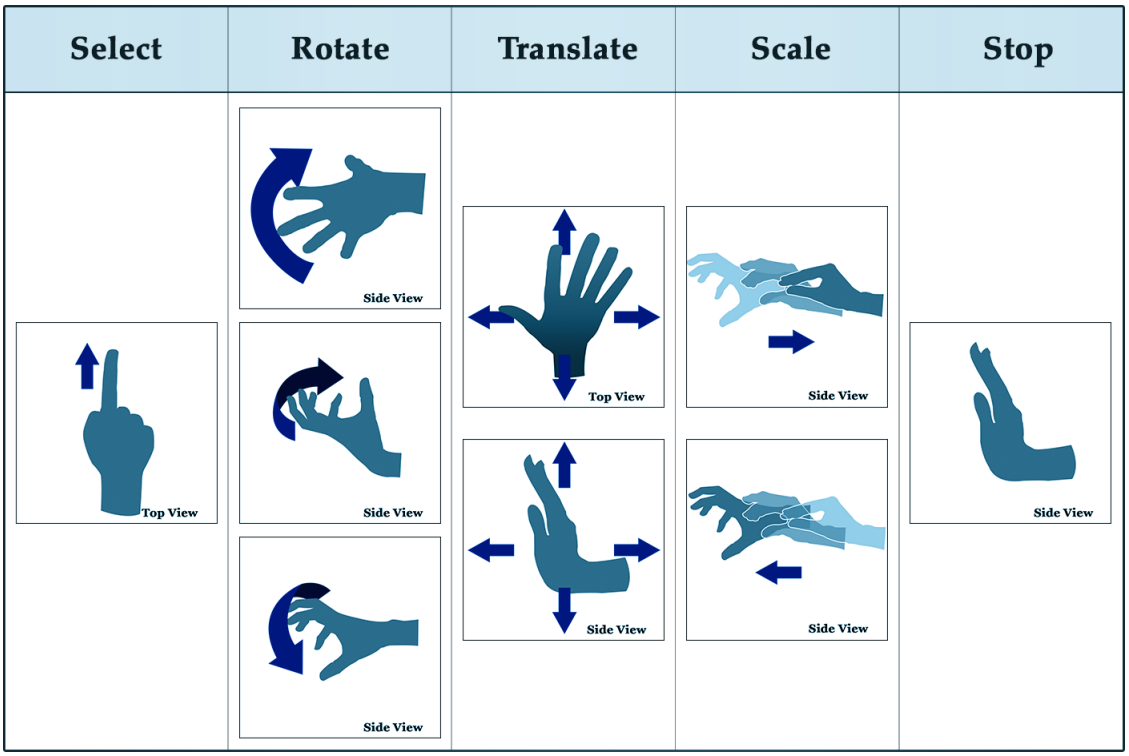


Figure 3.11. Gesture Vocabulary 1

3.2.4 Development of CubeNI

The CubeNI application was created from the integration of gestures in the CubeMK, both were developed using Unity. Mouse, Keyboard or Leap Motion events activate methods and properties from Unity, this last is responsible for the visual production of the aforementioned applications. Observe the UML use case diagram in Figure 3.14.

First it will be explained how the Unity performs rotation, translation and scaling of objects, then how is the logic behind Select and Stop, and finally it will be explained how, in each application, mouse, keyboard or Leap Motion events are treated.

For the rotation and scaling of the object in the scene the author used, in both applications, respectively, transform.Rotate() method and property transform.localScale from Unity's library. For the translation of the object in CubeNI, the author used transform.Translate() method, and in CubeMK the property transform.position. To Select and Stop, both applications have the Boolean

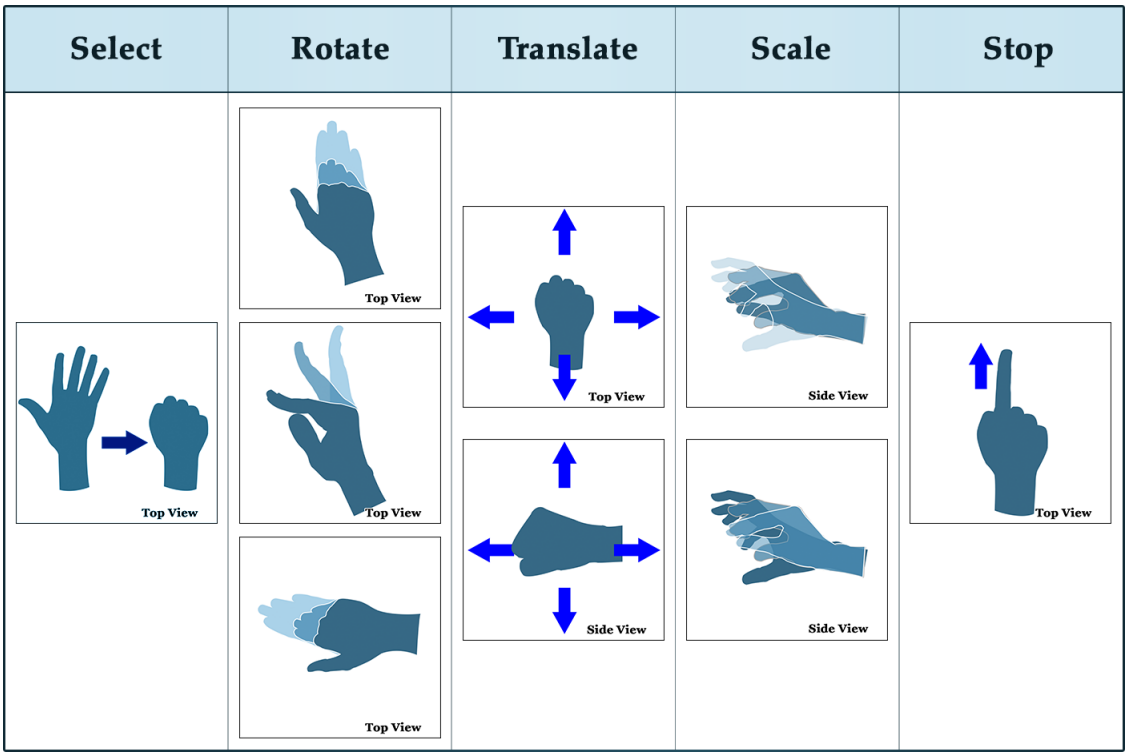


Figure 3.12. Gesture Vocabulary 2

variable “selected”, that when true indicates that the cube object is selected, and when false indicates that you want no interaction with the object, thus ignoring any mouse and keyboard event or Leap Motion‘s. The Unity has a Start() method that is called once at the beginning of the application, and an Update() method that is called with each new frame, every Update() checks the status of “selected”.

In CubeMK, to select the cube, one must click it. Unity then captures the mouse event OnMouseDown() and makes the value of the variable “selected” true. In CubeNI the Leap Motion captures the TYPESCREENTAP gesture that is pre-defined in its library and then follows by modifying the value of the variable “selected” to true.

In CubeMK, to rotate the cube, you must click and drag with the first mouse button, the Unity then captures the mouse event GetMouseButton(0) and starts to use the location of the mouse pointer as a parameter in transform.Rotate(), while the first button remains pressed. In CubeNI, to rotate, the Leap Motion analyzes the probability of rotation relative to the previous frame (Rotation-

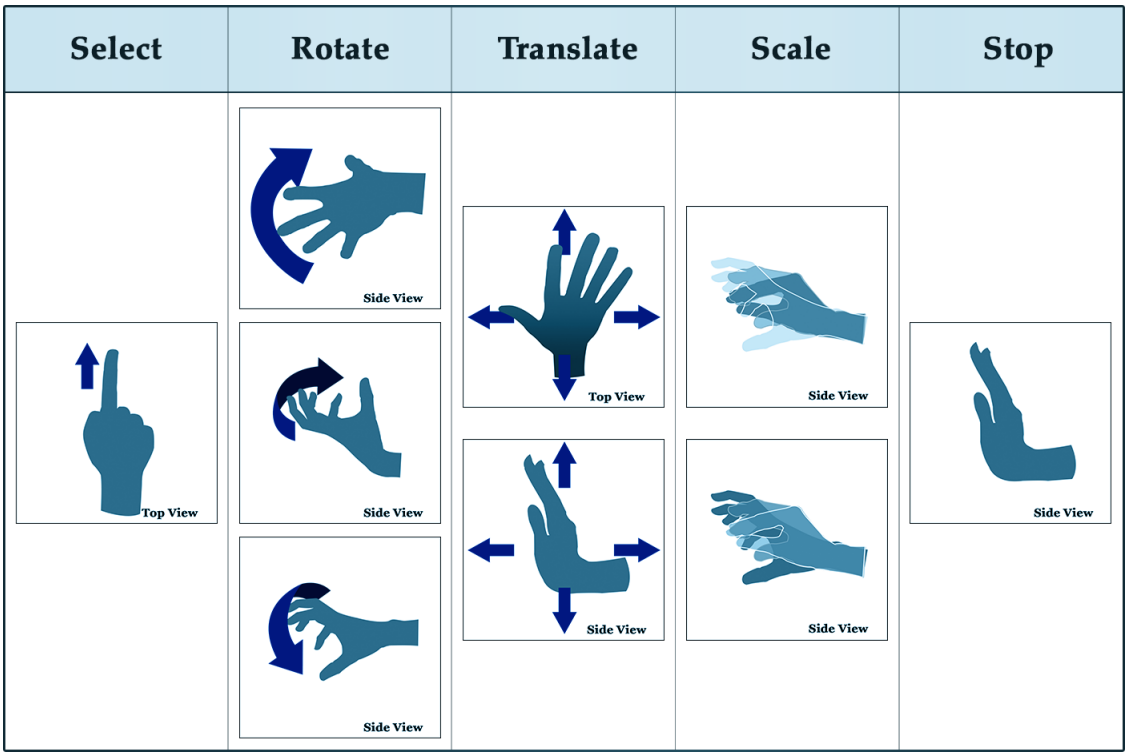


Figure 3.13. Template of Final Gestures

Probability (previousframe)) and the difference in the value of the hand axis from the previous frame to the current (RotationAxis (previousframe)), which is used as a parameter in transform.Rotate(). In both applications, for the above mentioned to occur, it is necessary that "selected" be true.

In CubeMK, to translate the cube, you must click with the second mouse button and drag. Then Unity captures the mouse event GetMouseButton(1) and proceeds to update the cube current position (currentPosition) relative location of the mouse pointer, while the second button remains pressed. In CubeNI, to translate, the Leap Motion analyzes the probability of translation from the previous frame (TranslationProbability(previousframe)) and how close the hand position is to a fist (GrabStrength), if the value of this variable is low (hand is open), then Unity begins to use the hand position as a parameter in transform.Translate( ). In both applications, for the above mentioned to occur, it is necessary that "selected" be true.

In CubeMK, to scale the cube, you must move the scroll wheel of the mouse, then Unity captures the GetAxis("Mouse scrollwheel") event and proceeds to

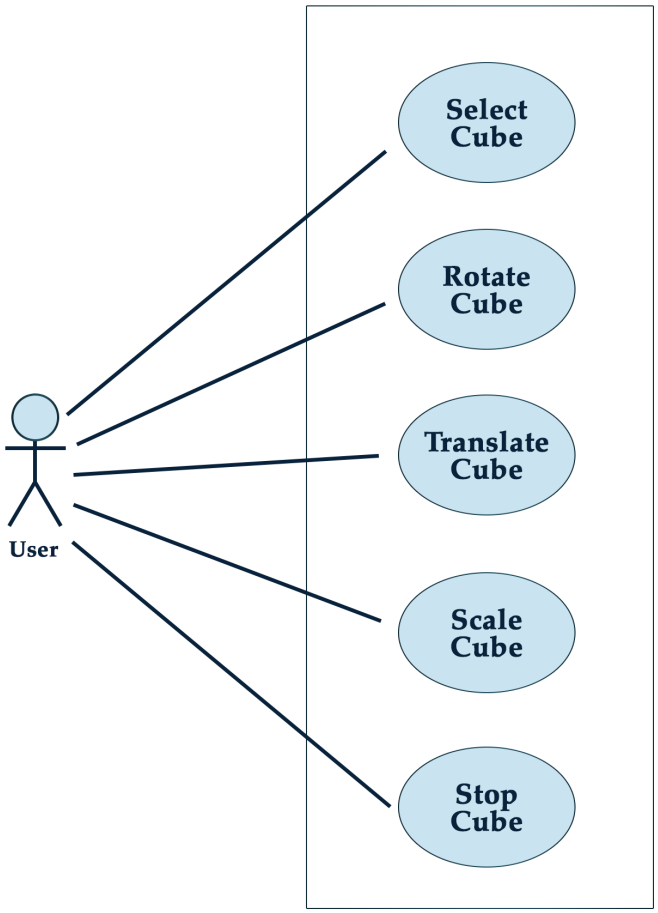


Figure 3.14. UML use case diagram of CUBENI

update the cube‘s localscale, in relation to the direction in which the scroll wheel is being scrolled to, while it continues to move. In CubeNI, to scale, the Leap Motion analyzes the likelihood of escalation from the previous frame (ScaleProbability (previousframe)) and the proximity of fingers, between themselves, in relation to the previous frame (ScaleFactor (previousframe)). The value “1” indicates no scaling, values above “1” indicates an increase in scale, and values between “0” and “1”, reduction. Unity then passes these values to modify the localScale. In both applications, for the above mentioned to occur, it is necessary that “selected” be true.

In CubeMK, to stop the cube, one must press the keyboard space bar, then Unity captures the GetKeyDown(“space”) keyboard event and makes the value of the variable “selected” false. In CubeNI, the Leap Motion check how close

your hand is to be a fist (GrabStrength), if the value of this variable is low (open hand) then the Unity counts how long the hand stoop open and motionless, should it be approximately two seconds or more, Unity makes the value of the variable "selected" false.

3.2.5 Evaluation of User Experience

In this work it was opted for the evaluation of the user's experience that analyzes in the broadest way the interaction user-system and comprehends the most intimate relation of the user and his sensations with the application.

For [43], the experience of the user is characterized by the experience that the user acquires when he interacts with a product. According to the author, the user's experience is defined as being his emotions and expectations and his relation with other people and with the context of use, where it is highlighted affective aspects and personal experiences. In a complementary way [44] defines the experience of user as: "an emergent change in the concept of usability, where the objective is not limited to enhance the performance of the interaction user-efficiency and ease of learning, but he tries to solve the strategic problem of the utility of the product and the psychological problem of pleasure and fun of its use".

According to the Microsoft user Experience guide, the good experience of the user happens when one is able to create pleasant, positive and satisfactory sensations, on the use of a system, being the fidelity of the user to the system the achievement of a good experience. In this way, the evaluation of the user's experience gives us the needed objects to identify positive and negative aspects on the interaction process of the user with the system, leaning on the very own user's perspective.

The objective of evaluating the user's experience is to validate the developed GDP, the template of the final vocabulary of gestures, as also the vocabulary itself. To this end, the tests were executed with 20 voluntaries, professors and undergraduate students of Computer Engineering and Computer Science courses from Federal University of Paraíba, Brazil.

The evaluation of the user's experience used the tool AttrakDiff [45]. This

tool has the objective of evaluate the perception of the qualities of products or systems on the user's perspective. In this way, it is possible to study the aspects of the user's experience in an applied context that portrays not only pragmatic aspects, but also impressions, reactions and sensations of user. In this approach not only the functional aspects of the software are important, but also the interaction characterizes itself, for example, stimulating, it leans and allows the creativity and what kind of symbolic value it possesses.

The AttrakDiff questionnaire gauges how attractive a product is in terms of usability and appearance [45]. The theoretical work model has been researched and tested in several studies [46, 47, 48, 49]. The AttrakDiff questionnaires, observes four characteristics on the product evaluation: Pragmatic Quality (PQ), Hedonic Quality - Stimulus (HQ-S), Hedonic Quality - Identity (HQ-I), and Attractiveness (ATT). The PQ refers to usability of the product: whether or not the users are being able to reach their objectives using the product. the HQ-S worries about how innovative and interesting is the product and if it has content, functions, interaction and arousing styles of presentation, the HQ-I indicates how the analyzed product allows the user to identify with it, and at last, the ATT describes the global value of the product based on the quality of perception.

The voluntaries tested both applications CubeMK, now with textures (Figure 3.15), and CubeNI (Figure 3.15). Notice that the appearance of both applications are identical, for this way the only difference between them is the form of interaction, and the voluntaries evaluated them using questionnaires generated by the AttrakDiff. Each questionnaire presented 28 pairs of opposing ideas, such as complicated-simple, ugly-beautiful, human-technological, and 7 possible levels of choice between each word from the pairs, thus it was asked to each voluntary to fill two forms, being one for each application.

The comparative results of the applications can be observed in Figures 3.16, 3.17 and 3.18.

Fig. 8 shows in full the word pairs used in the description of each questionnaire. Note that the word pairs are grouped according to the four characteristics mentioned before; Pragmatism (PQ), Identity (HQ-I), Stimulation (HQ-S) and

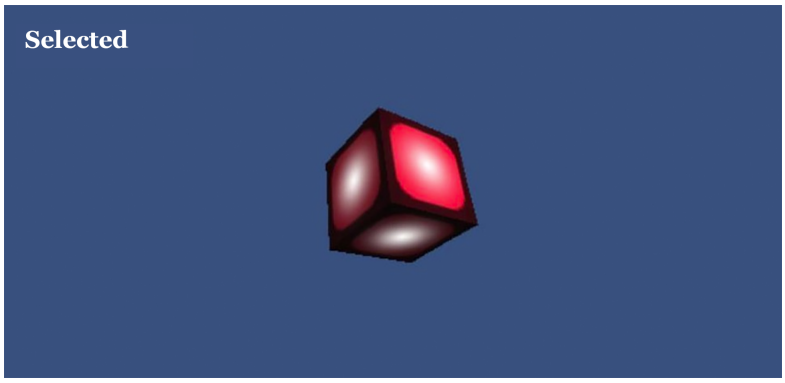


Figure 3.15. Image of the Cube that suffers the selection, rotation, translation, scaling and stop actions. The same Cube is used in two applications with mouse and keyboard (CubeMK) and gestural (CubeNI) interfaces.

attractiveness(ATT).

It is important to notice that there are two lines in Fig. 8, the blue represents the CubeNI and orange the CubeMK. For each pair of words there is a point on each line. Each point represents the average result for the answer of the 20 volunteers for that pair of words, for each application. Therefore, it is possible to compare the results of the applications in question.

In terms of Pragmatism (PQ), note that for the first couple of words, "technical-human," the CubeMK is classified as "technical" while CubeNI got more points on the "human" side, it reflects well the nature of the interactions on the two applications, since the natural interaction places a high value human side of human-computer interaction, and interaction via mouse and keyboard prioritizes the task, being more technical.

In terms of identity (HQ-I), knowing that "separates me - brings me closer" want to give the idea of "separates me from people - brings me closer to people" note that the CubeMK was classified approximately neutral in this regard, while the CubeNI received a moderately better score, indicating that the use of this interface gives an idea of interpersonal connection, a characteristic that relates to the human side of the natural interaction.

In terms of Stimulus (HQ-S), note that in "dull -captivating" the CubeNI was rated more captivating than the CubeMK, and not only in this pair, but in all

pairs of the HQ-S group, the CubeNI obtained better results than the CubeMK, thus being considered overall as giving a more stimulating experience for the user.

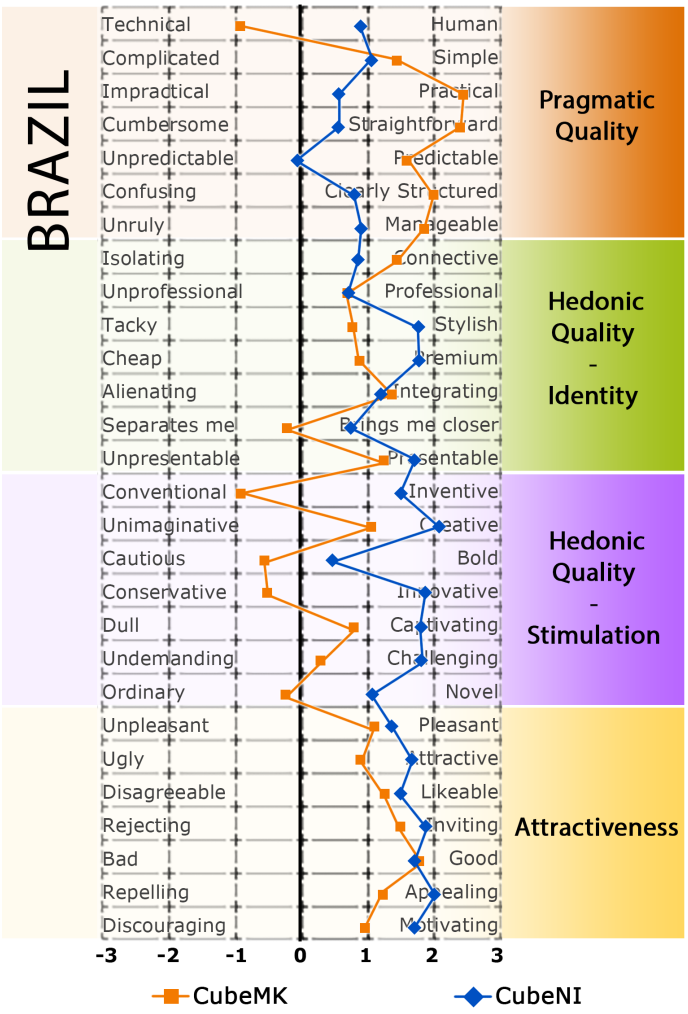


Figure 3.16. Comparison between CubeMK and CubeNI using pairs of words from AttrakDiff

In terms of attractiveness (ATT), observe the pair “bad - good”. It is interesting to note that there was minimal difference in the results of applications, both CubeMK when CubeNI were practically considered equally good, which reflects the conflict of volunteers when using an interface that is familiar to them comfortably, in contrast to an exciting new interface that is displayed on the CubeNI.

Figure 3.17 shows the average of the results shown in Figure 3.16. It is worth noticing that the CubeNI presents better overall results in the categories

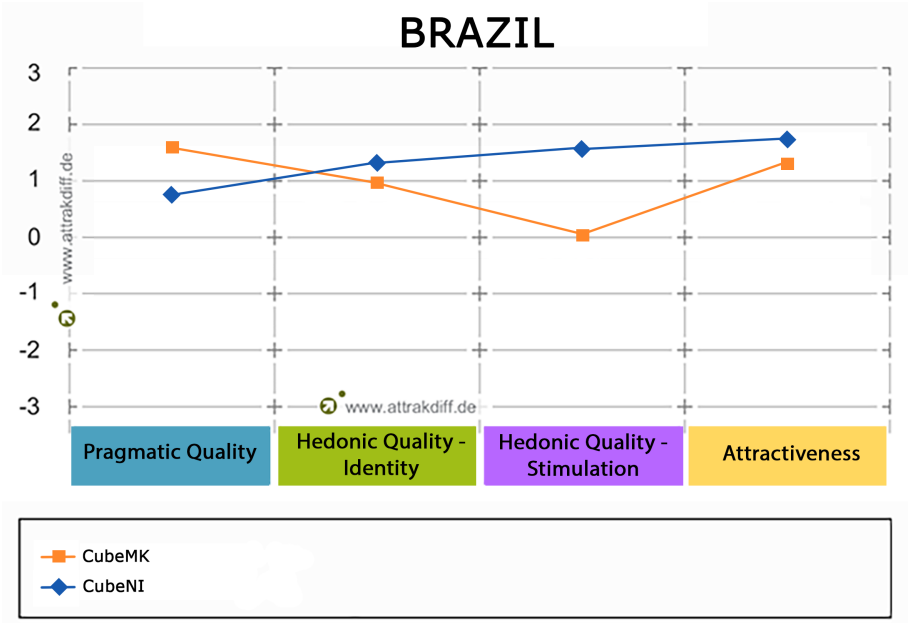


Figure 3.17. Average results

Identity (HQ-I), Stimulation (HQ-S) and attractiveness (ATT), while CubeMK is better in Pragmatism Question (PQ), one of the possible reasons for this is that the use of mouse and keyboard makes the system task-oriented, because it provides more precision in its handling, also because the average user is already acquainted with this type of system and, although it requires training, the constant use in everyday life becomes comfortable enough for the user to focus on the goal and not in the means. However, one can see a great potential in the use of Natural Interaction devices, such as the Leap Motion in the CubeNI: despite the gestural interface failing to overcome the popularity of the mouse and keyboard interface in the item PQ, the gestural interface already shows superior results in HQ-S, ATT and HQ-I. That is, more stimulating, attractive and the user can identify better with the gestural interface. It is expected then that with the constantly evolving and growing integration of these devices in everyday life, it can also improve its pragmatism, making it an excellent choice of interaction model.

Figure 3.18 depicts a focused display between pragmatism and hedonism, therefore showing in a more simplified way that despite CubeMK being more pragmatic, the CubeNI is closer to the ideal result.



Figure 3.18. View of the final test results

Therefore, it was possible to perceive that the CubeMK is more focused on task execution than on the experience with the user, and the CubeNI gives a good importance to user interaction, leaving something to be desired in guiding the task, this is due to lower accuracy of the NI devices compared to the mouse and keyboard, but those are constantly evolving while mouse and keyboard are stagnant in their dimension of interaction. It is expected that with every technological advance this deficiency in the pragmatism of NI devices will be reduced until remedied.

3.3 Conclusion

The main objective of this research was to investigate the user interface development process based on gestures, this style of interaction, has a wide range of potential applications, far beyond those described in this work.

To achieve the proposed objective a literature review was initially conducted in search of works that addressed the development and design of gestures. From there the author saw that little importance was given to this matter: several works which had a gestural interface were observed, but without performing any process of gesture development. When this occurred, the result was noticeable among the users, which commented on the possibility of improvement of gestures, or even the degree of difficulty or exhaustion when performing them repeatedly.

This research presented a Gesture Development Process to guide designers through the development of user interfaces based on gestures. The documentation produced throughout this Gesture Development Process enables collaborative initiatives and support reuse of previous efforts. Additionally, a gestural interface was developed in Brazil, using the proposed Process. Named CubeNI (Natural Interaction), a prototype was implemented using the Leap Motion device.

The main objective of the Gesture Develop Process is to output a satisfactory Gesture Vocabulary, so that they can be useful for specific system functions. Therefore, the author evaluates and validates that resulting Gesture Vocabulary, in the form of the CubeNI prototype, to indirectly validate the Gesture Development Process, similar to an outcome-based evaluation [50]. The resulting application CubeNI was validated and evaluated in Brazil.

3.3.1 Contributions

The contributions of the work presented in this chapter were:

- Presentation of an adaptive Gesture Development Process model and artifacts to assist in developing user interfaces based on gesture, guiding the designers in choosing the most appropriate gestures for their applications and enabling collaboration and reuse of past results;
- Through the proposed model, the development of a vocabulary of gestures aimed at the three-dimensional context for the functions “Select”, “Rotate”, “Translate”, “Scale” and “Stop”;

- The deployment of the vocabulary of gestures through the development and evaluation of two pilot applications, one with gestural interfaces and other with mouse and keyboard.

Chapter 4

Multicultural Gestural Perceptions

4.1 Introduction

Evaluating gestural interfaces is a complex task because it involves many nuances of perceptions that can easily change from user to user. This work proposes an investigation and discussion about how to improve gestural interfaces by presenting a user experience evaluation where opinions about a mouse interface and a gesture interface were compared using metrics that tried to capture the abstract perceptions of the subjects. The gesture interface application's strong and weak qualities were identified and analyzed by volunteer participants in different cultures. Understanding how gesture interfaces are perceived in different cultures can be helpful to contribute to the effectiveness of this type of interface. This understanding can also help to reuse the gestures developed for one user group, with another, saving the effort of performing another Gesture Development Process.

4.2 Overview of experiment

Gestural interfaces should ideally have the ability to always determine when the user was moving naturally, just scratching his nose, waving to a friend, or really intended to access a function of the system. It should be able to adapt its recognition algorithm to every different person in order to provide accuracy and better anticipate each user's needs. Gestural interfaces should be able to seamlessly blend into our lives to facilitate tasks, assist in daily life, or even

encourage healthier pastimes.

In order to achieve this level of perception, challenges have to be addressed both on the human side and on the technological side. This work focuses on exploring the user's perception of gestural interfaces and comparing it to those of a conventional mouse interface. In order to do that, the author sets up an experiment using two applications: one with a gestural interface using leap motion, the other using a mouse and keyboard. Both applications had the same system functions; users could Select, Rotate, Translate, Scale and Deselect a red cube on the screen, the only difference was the input mode. Subjects from two different countries could test both applications, sometimes making remarks about the interaction. The gestures for the gestural interface used were developed using participants from only one of the countries. To try to capture more abstract nuances of user's perceptions about both systems, subjects were asked to answer questionnaires provided the AttrakDiff tool [45].

The AttrakDiff is a 28-item semantic differential tool used to measure the perception of two broad qualities: Hedonic quality and pragmatic quality [51, 52]. Hedonic quality refers to the fulfillment of psychological needs and is represented by two subdivisions: Identification and Stimulation. Pragmatic quality refers to a product's perceived potential to support relevant task achievement. In addition, AttrakDiff also measures general appeal. Appeal is encapsulated as the consequences of pragmatic and hedonic quality perception.

Two universities provided available resources to carry out the study: the Federal University of Paraíba in Brazil and Osaka University in Japan. The choice of places for the experiments was motivated due to the different access levels of the general population regarding technology in daily life. Japan was placed 10th in a global ranking that assesses nations on their level of development in information and communication technology [53], as a result of high levels of investment in ICT infrastructure, high-quality networks, and high levels of take-up of services by consumers. The same study places Brazil on the 66th position, showing a considerable gap between both countries, which is also true in previous assessments [54]. Universities are a place were individuals from various financial backgrounds, beliefs, and views can meet and socialize with

each other; this environment seemed beneficial to analyze how different technological familiarization levels impacts the user experience of gestural interfaces.

The following sections describe the experiment procedures, participant groups, and data collection in detail and then analyze the results obtained from the study. Confidence intervals and a 2-tailed Student's Paired T-test were conducted on the average results for each word pair from the questionnaires.

4.3 Materials and Human Resources

The following text describes the participants that volunteered for the experiment, the applications, equipment, and questionnaires that were used, and how the experiment was setup.

4.3.1 Participants

For each country, 20 subjects took part in the experiment.

- In Brazil (age: min=19, max=43, $\bar{x}$=23.45, SD=5.5), the subjects were Brazilian students of the Federal University of Paraíba. No one had previous experience with gesture interfaces or knew about them.
- In Japan (age: min=21, max=51, $\bar{x}$=25.95, SD=7.3), the participants were Japanese students and staff of Osaka University. Only one participant had previous experience with gesture interfaces, but 33% knew about it. In both countries half of the participants were female. The experiment was conducted individually with each participant.

4.3.2 Applications

Two applications were used: CubeNI (Natural Interaction) and CubeMK (Mouse and Keyboard). The gesture set for CubeNI was defined using a Gesture Development Process [55] presented on Chapter 3, and they can be observed in Figure 4.1. The experiment in Brazil was originally executed with the objective of verifying and validating this methodology. In CubeNI and CubeMK, the user could interact with a red 3D cube on the screen. After selecting it, the cube

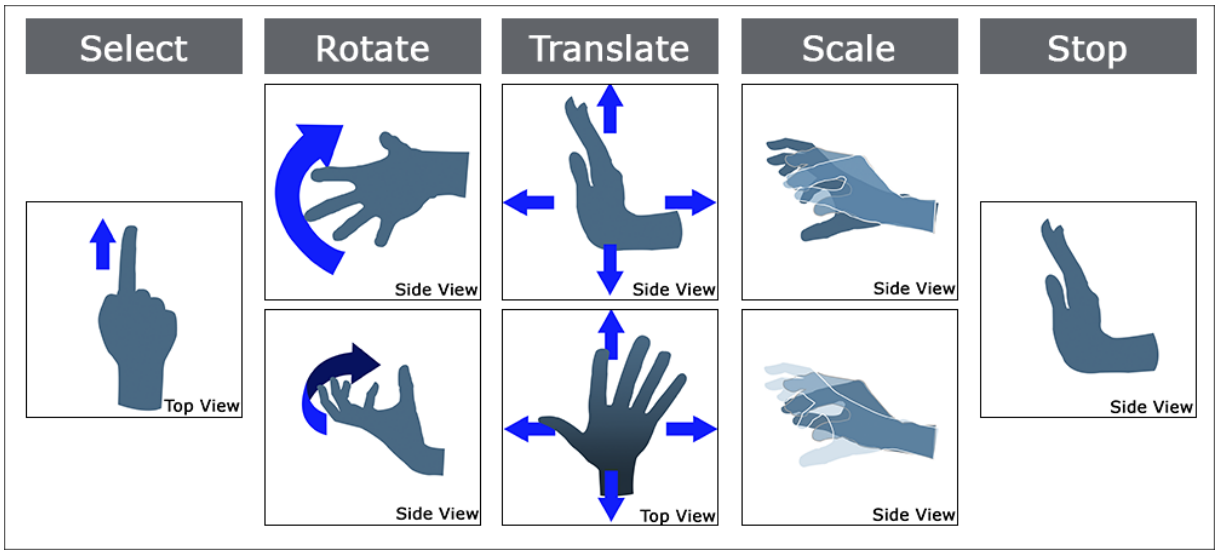


Figure 4.1. CubeNI's Gestures

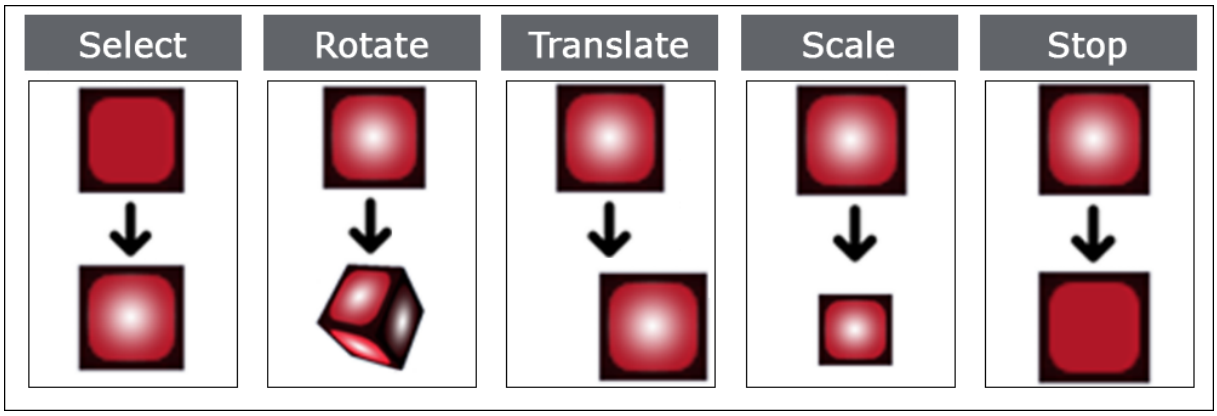


Figure 4.2. CubeNI and CubeMK's possible interactions with the red cube.

could be rotated, translated, or scaled. Additionally, all the interaction with the cube could be stopped by deselecting the cube. Figure 4.2 shows all possible interactions with the red cube on both applications. CubeNI utilized a gestural interface through Leap Motion, while CubeMK utilized a mouse and keyboard interface. The keyboard was only used to deselect the cube.

4.3.3 Tangible Resources

The Leap Motion controller was chosen as the gesture capture device. A wireless mouse and a laptop with a 14-inch display at a 1920 x 1080 resolution was used to conduct the experiment. All participants completed the experiment sit-

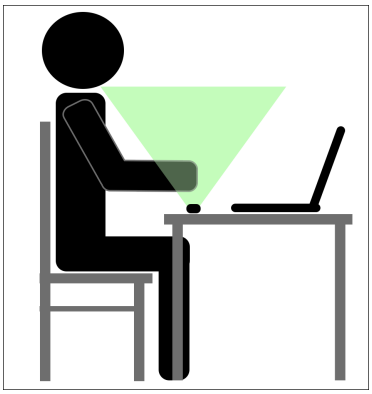


Figure 4.3. Experiment set-up

ting on a table with the devices adjusted at a comfortable distance in front of them, as shown in Figure 4.3. The provided questionnaires were obtained using the AttrakDiff tool.

4.3.4 Questionnaires

The author also used the AttrakDiff questionnaire [45] to gauge the multicultural perception. As mentioned in chapter 3, the questionnaires used consisted of 28 seven-step items whose poles are opposing adjectives (e.g. “confusing - clear”, “unusual - ordinary”, “good - bad”). Each set of adjective items is ordered into a scale of intensity. The middle values of an item group create a scale value for PQ, HQ and ATT.

The AttrakDiff tool provides an English and German version of its questionnaire, so in order to ensure that the subjects had full understanding of what was asked, native language translations accompanied the official English version of the questionnaire on both countries.

4.4 Conducting the Experiment

The experiment described here explores what users’ preferences and perceptions were when using different interfaces for performing the same tasks.

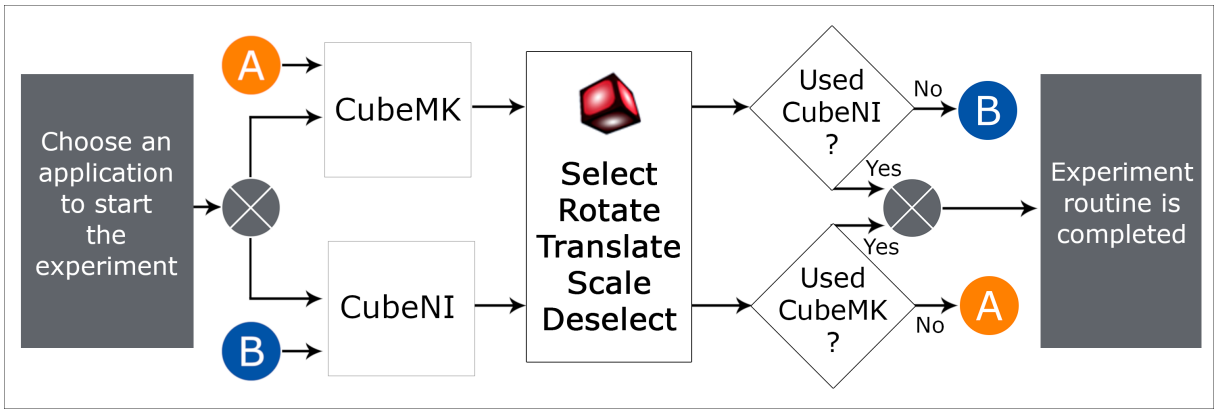


Figure 4.4. Experiment's Routine

4.4.1 Routine

One of the two applications (CubeNI or CubeMK) was chosen at random to start the experiment. The participant was then asked to Select, Rotate, Translate, Scale, and Deselect the cube. After all tasks were completed, the remaining application was then started and the same tasks repeated (Figure 4.4).

In both countries, half of the subjects started the experiment with the gesture application, CubeNI, and the other half with the mouse application, CubeMK, to offset effects introduced by the order of the procedures. After testing both applications the participants were given two copies of the AttrakDiff questionnaire; one for each application.

4.4.2 Results

In order to capture users' perception of the product, AttrakDiff uses pairs of opposite adjectives as a method of investigation. These adjective-pairs make a collation of the evaluation dimensions possible. The following product dimensions are evaluated: The Pragmatic Quality (PQ) describes the usability of a product and indicates how successful users are in achieving their goals using the product. The Hedonic Quality - Stimulation (HQ-S) dimension indicates to what extent the product is novel and interesting, considering stimulating functions, content, interaction, and presentation styles. The Hedonic Quality-Identity (HQ-I) dimension indicates to what extent the product allows the user

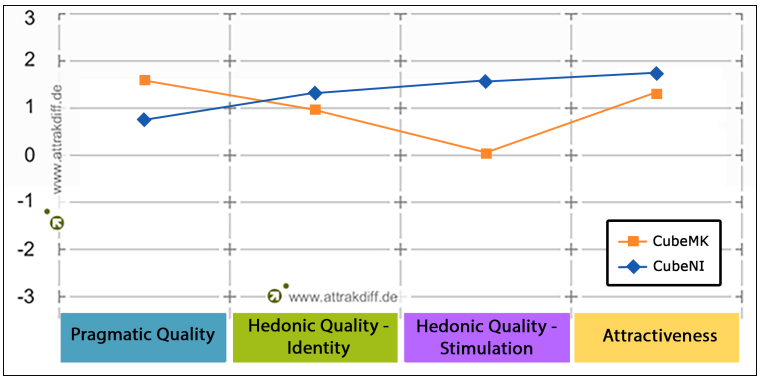


Figure 4.5. Brazilian Experiment's Diagram of Average Values

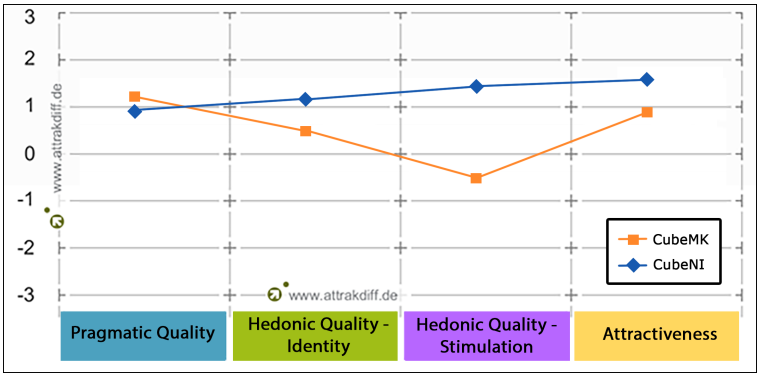


Figure 4.6. Japanese Experiment's Diagram of Average Values

to identify with it. The Attractiveness (ATT) dimension describes a global value of the product based on the quality perception. Hedonic and pragmatic qualities are independent of one another, and contribute equally to the rating of attractiveness.

Those four dimensions are depicted in three graphics: Diagram of Average Values, Description of Word-Pairs, and Portfolio of Results. The Diagram of Average Values combines the means for each of the four dimensions evaluated. In both countries' Diagram of Average Values (Figures 4.5 and 4.6) it can be observed that Brazil's and Japan's perceptions of both types of interface were similar: CubeMK was considered more pragmatic, meaning that it permitted the user to achieve his objectives more easily, while the CubeNI was deemed more stimulating, attractive, and the user could identify himself/herself more easily with the product when using this type of interface.

The Description of Word-Pairs shows the average of the questionnaire an-

swers by the 20 volunteers for each word pairs, in each country. In Brazil (Figure 4.7), the biggest differences between CubeMK and CubeNI results were that the mouse application was considered much more Practical and Straightforward whereas gesture application was regarded as much more Human and Innovative. However, both applications were considered to be equally Good and Professional to the Brazilian subjects.

In Japan (Figure 4.8), considering all major divergence points between both applications' average scores, CubeNI's marks were closer to the best possible scores than CubeMK's. Also, CubeNI was especially considered to be highly Innovative and a Novelty (as opposed to Ordinary). As for the converging points, CubeMK and CubeNI were regarded as equally Pleasant and Simple.

When comparing CubeMK's results between both countries the greatest significant difference found was that in Brazil, it was judged to be Creative whereas in Japan it was considered Unimaginative. On both countries it was found to be similarly Pleasant, Simple, Predictable, and Manageable.

When comparing CubeNI's scores in Brazil and Japan, a big difference in opinion was that the Brazilian subjects found the gesture application to be Human, whereas in Japan it was deemed Technical; Another diverging point was that Japanese subjects found the gestural interface to be Undemanding, whereas Brazilians thought it Challenging. The common opinion was that CubeNI was considered equally Attractive and similarly Integrating and Simple.

In the Portfolio of Results, the values of HQ are represented on the vertical axis (low values at the bottom). The horizontal axis represents the value of the PQ (low values at the left). Depending on the dimension's values, the product will lie in one or more of these presented regions. The bigger the confidence rectangle, the less sure one can be to which region it belongs. A small confidence rectangle is more informative since investigation results in that area are less coincidental, making them more reliable. The confidence rectangle essentially shows if the users are in agreement in their evaluation of the product. The bigger the confidence rectangle, the more variable the evaluation ratings.

By analyzing both countries' portfolios of results, the author observed that the Brazil's CubeMK confidence rectangle (Figure 4.9A) is wider on the HQ

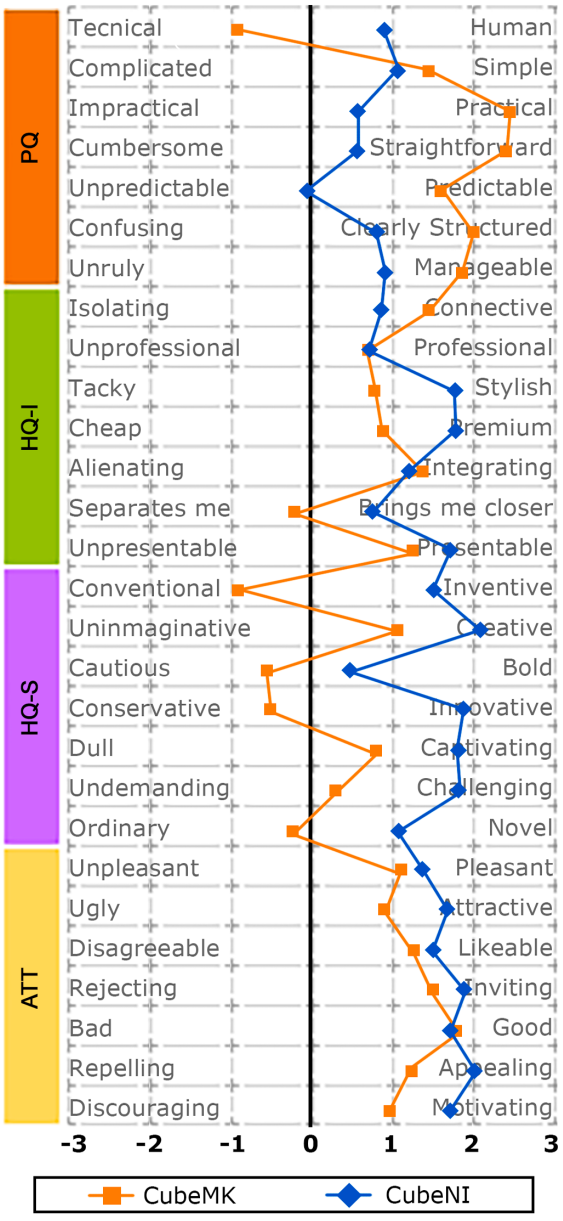


Figure 4.7. Brazil's Description of Word-Pairs

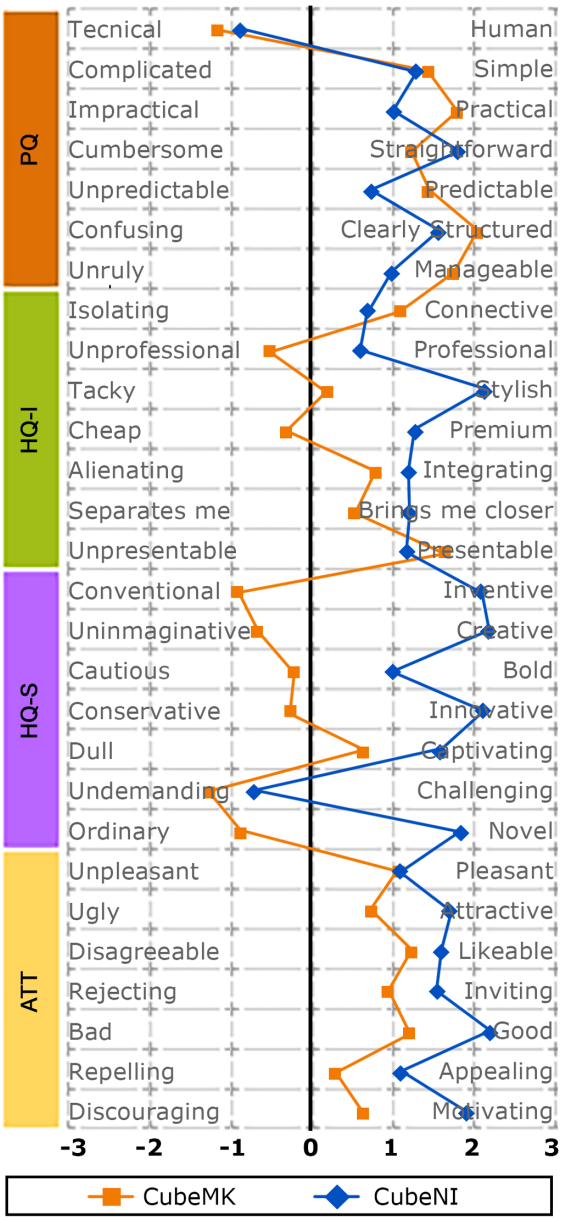


Figure 4.8. Japan's Description of Word-Pairs

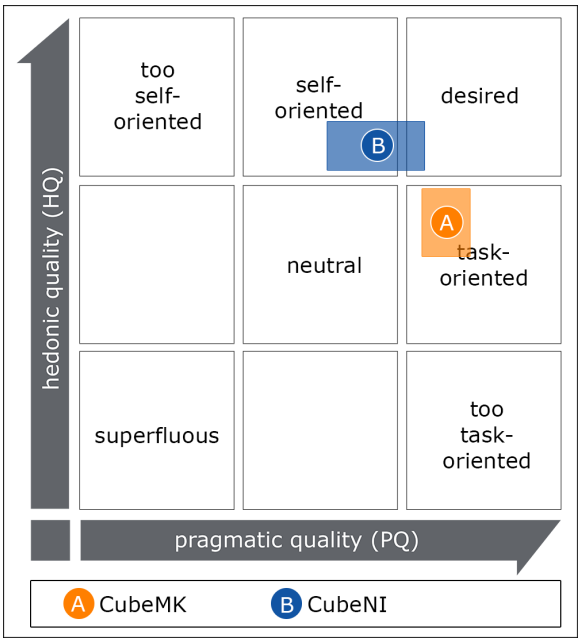


Figure 4.9. Brazil's Portfolio of Results

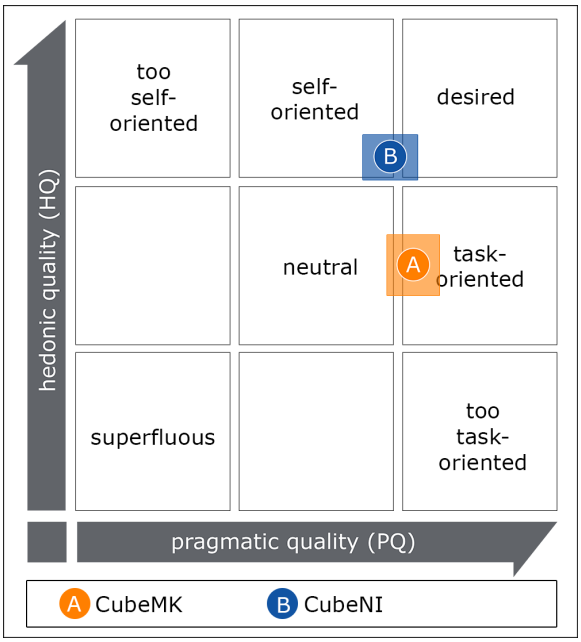


Figure 4.10. Japan's Portfolio of Results

than the PQ axis, suggesting that the subjects in Brazil had a more homogeneous opinion regarding CubeNI's PQ dimension. As for Brazil's CubeNI confidence rectangle (Figure 4.9B), the opposite occurred; it was wider on the PQ axis than the HQ axis, suggesting that the subjects in Brazil had a more homogeneous opinion regarding CubeNI's HQ dimension. This portfolio considers the balance between self-oriented and task-oriented features; the quadrant marked "desired" represents values that would be ideal in this case. The Brazil's CubeMK confidence rectangle was deemed task-oriented, and CubeNI confidence rectangle fell around 33% inside the desired quadrant, although on its most uncertain axis.

Japan's CubeMK confidence rectangle (Figure 4.10A) is a little wider on the HQ than the PQ axis, suggesting that the subjects in Japan had a more homogeneous opinion regarding CubeNI's PQ dimension. Japan's CubeNI confidence rectangle (Figure 4.10B) is considerably smaller on both the HQ and PQ axis, suggesting that the subjects in Japan had a more homogeneous opinion regarding CubeMK's PQ and HQ dimensions. The Japan's CubeMK confidence rectangle was also deemed mostly task-oriented although it also leaned towards the neutral quadrant. Japan's CubeNI confidence rectangle fell around 20% inside

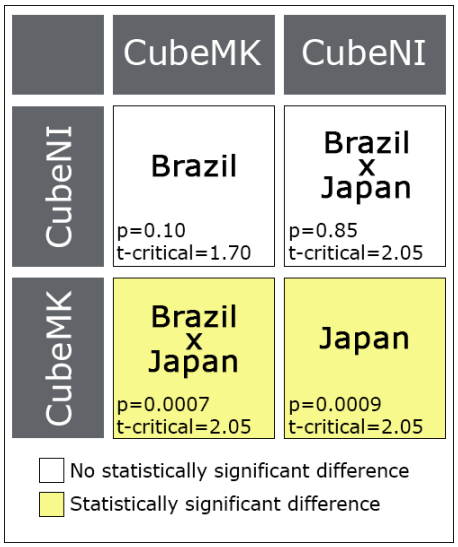


Figure 4.11. Student's Paired T-test overall results between Brazil and Japan.

the desired quadrant, where the opinions of the Japanese subjects were clearly more homogeneous than the Brazilian subjects.

A 2-tailed Student's Paired T-test (Figure 4.11) was conducted on the results obtained from the questionnaires. A significance level of 5% was considered. In Brazil, CubeMK and CubeNI were compared (p=0.10, t-critical=1.70) and no statistically significant differences were found. Additionally, a 2-tailed Student's Paired T-test between the results of each of the 4 qualities groups (PQ, HQ-I, HQ-S, ATT) was also conducted (Figure 4.12). For PQ (p=0.14, t-critical=2.44) and HQ-I (p=0.19, t-critical=2.44) there were no statistically significant differences found, but for HQ-S (p=0.0006, t-critical=2.44) and ATT (p=0.009, t-critical=2.44) there were statistically significant differences between subjective metrics for both applications in Brazil.

In Japan, a statistically significant difference was found between CubeMK and CubeNI's results (p=0.0009, t-critical=2.05). The results for each quality group was compared between both applications; here it was observed that in the PQ (p=0.20, t-critical=2.44) and HQ-I (p=0.10, t-critical=2.44) there were no statistically significant differences, but for the HQ-S (p=0.002, t-critical=2.44) and ATT (p=0.004, t-critical=2.44) there were statistically significant differences found between both applications in Japan.

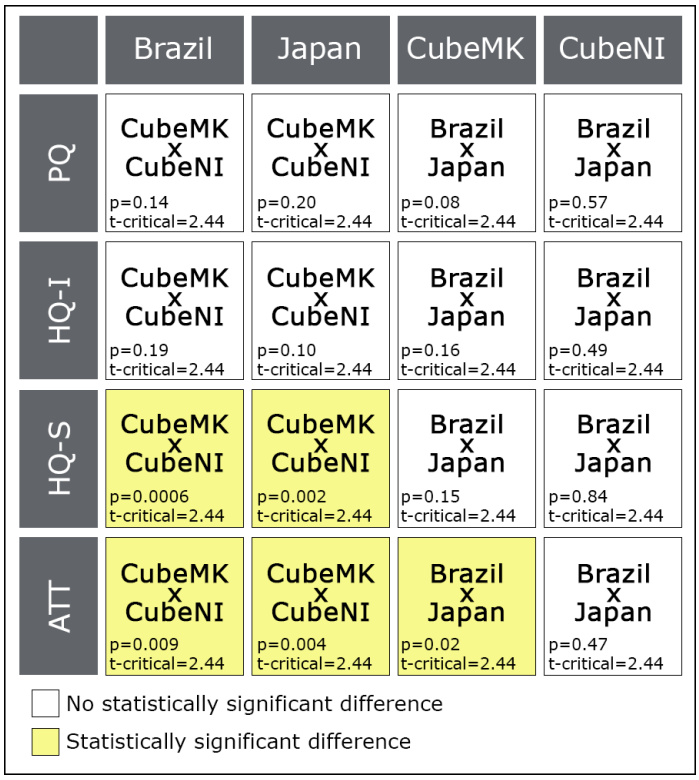


Figure 4.12. Student's Paired T-test results between Brazil and Japan, considering every possible comparison regarding the four qualities (PQ, HQ-I, HQ-S and ATT).

When comparing CubeMK in Brazil and Japan, the results (p=0.0007, t-critical=2.05) show a statistically significant difference. For the group of qualities PQ, CubeMK's results in Brazil and Japan were compared and the outcome (p=0.08, t-critical=2.44) shows that there were no statistically significant differences found. The same goes for HQ-I (p=0.16, t-critical=2.44) and the HQ-S (p=0.15, t-critical=2.44), but for ATT (p=0.02, t-critical=2.44) there was a statistically significant difference found.

The scores (p=0.85, t-critical=2.05) for CubeNI's results in Brazil and Japan showed no statistically significant difference. For PQ (p=0.57, t-critical=2.44), HQ-I (p=0.49, t-critical=2.44), HQ-S (p=0.84, t-critical=2.44) and ATT (p=0.47, t-critical=2.44) CubeNI's results shows that no statistically significant differences were found.

4.5 Discussion

In a utopia, any system should have a perfect understanding of what its users want, when they want and how they would like to get it done, preferably requiring minimal effort by the user. In order to achieve that its imperative to have accurate motion tracking, fast learning curve for interactions (intuitive gestures), understand user's individual preferences, distinguish between idle and action-requesting motion. Gesture interfaces must adapt to how individuals express unique practices, so in order to better understand it this study was conducted.

During the course of the tests, all subjects were excited to take part in the experiment, some had used kinetic previously, and one Leap Motion, but the great majority had never considered gesture interfaces for tasks other then entertainment.

Subjects encountered difficulties on understanding how to use the questionnaires properly, all subjects were instructed to try to give a spontaneous answer when matching between each word-pair, even if failing to see how some word-pairs applied to the applications used. Overall this difficulty was expected, given that this was a subjective study, although it was noted that this was a much bigger issue for the Japanese than Brazilian subjects.

The gesture interface system used in the experiments didn't have a perfect motion tracking accuracy. Therefore, there were times when the system couldn't understand the gesture being performed, on those cases the lack of satisfactory feedback promoted frustration and dissatisfaction on the users. Comments were made that the interface should try give feedback to unrecognized gestures (e.g. "gesturing too fast, losing track of hands") in order to improve user satisfaction when better accuracy on gesture recognition is not possible.

One interesting result was that the application with gestural interaction was deemed more "Technical" than "Human" in Japan (Figure 4.8, Technical-Human): Japan's result for the mouse interface was very close together to its gestural one. The 95% confidence interval for all 20 Japanese answers for the Human-Technical word pair is as follows: for the CubeNI it varies from 2.2 to 4.0, for the CubeMK from 2.0 to 3.6. According to studies [53, 54], Japanese people are more familiarized with technology, compared to Brazilians, and while on the questionnaires those words are considered complete opposites, it seems that the perception in Japan was that they weren't. The perception of CubeNI being more "Technical" could come from the simple fact that both Leap Motion and Mouse are devices, therefore technical.

Overall, all subjects from both countries seemed very excited to try the gestural interface, but once this novelty wears off we need to think about building realistic standards for gestural interfaces. One very apparent challenge on this study was the metrics; how to choose or build a metric that can balance the scales when users from different backgrounds, preferences, interests, behaviors have different expectations and demands from a system. Subjectivity and abstract concepts are challenging, but on this study a point where there was major agreement was that the pragmatist quality of the gestural interface has a lot of room to improve in order to be embraced by the average user.

4.6 Conclusion

The aim of this work was to analyze how different technological familiarization levels impacts the perceived user experience of gestural interfaces. In order

to do that, the author sets up an experiment to compare a gestural interface (CubeNI) with a mouse interface (CubeMK). This experiment took place in Brazil and Japan, inside university environments. The AttrakDiff questionnaire was used to evaluate user experiences of both products.

As shown before, statistically significant differences were found only in two situations: between Brazil and Japan regarding CubeMK's results, and between CubeNI and CubeMK's results in Japan (Figure 4.11). As for the four qualities group (Figure 4.12), in Brazil, the comparison between CubeNI's and CubeMK's HQ-S and ATT qualities had statistically significant results. In Japan, the same goes for the comparison between CubeNI's and CubeMK's HQ-S and ATT qualities. Additionally, the comparison between Brazil-CubeMK's and Japan-CubeMK's ATT qualities also had statistically significant results.

In both countries the major advantage of the gestural interface was that it was considered much more stimulating to the users' senses than the mouse-based interface, and given the overall result the author estimates that the CubeNI could be used also in Japan with the same expected outcome as in Brazil, where the target-users are from. Therefore the Gesture Vocabulary could be reused, saving the effort or running another Gesture Development Process for similar system functions in Japan.

In both countries the disadvantage was that the pragmatic qualities of the gesture interface has yet to match the mouse's objectiveness. This suggests that there is still a need to improve the ease of use of gesture interfaces, so that the users' attention can focus entirely on task completion and not on interface usage. To that end it is recommended improving feedback when system cant recognize user's gestures.

The purpose of the Gesture Development Process is to output a satisfactory Gesture Vocabulary, for specific system functions. The author tries to extend previous efforts to new target users of different technological familiarization levels. The resulting application, CubeNI, was validated and evaluated in Japan. Having its results compared to the Brazilian counterpart. The results are similar among both groups. This suggests indirectly validation of the Gesture Vocabulary developed in the Brazilian context, in Japan.

Even if the new user-group cannot validate the Gesture Vocabulary, the documentation of the Gesture Development Process would still be valuable. The documentation tracks each step of the Gesture Development Process so it is possible to use any step of the process as a starting point for building either another Gesture Development Process, or having another elicitation study giving those gestures as options. Therefore, it is still viable to use some part of the process because it produces so much documentation, and to document so much is very time consuming. In the future this process can contribute with its production of relational data, aiding future efforts of non-human dependent Gesture Development Process.

Regarding gestural interfaces many paths could be taken to contribute to its development: Researching how to improve metrics for evaluating this type of interface would be welcomed, more studies in different countries can be beneficial to better understand what is globally expected from this type of interface, and research on gesture precision enhancement for future gestural interfaces could help to increase this type of interface's pragmatic qualities. To that end, during the course of tests, improvement on feedback was suggested when the system failed to recognize the gestures. Other possible continuation to this investigation course is measuring how users perform (time, effort, accuracy) when given more complex test routines, in addition to perception investigation, in order to balance subjectiveness with objectiveness.

4.6.1 Contributions

The contributions of the work presented in this chapter were:

- Validation of "select", "rotate", "translate", "scale", and "stop" gesture in Japan. Saving the effort of running another Gesture Development Process for similar system functions in Japan.
- Comparison of the multicultural gestural perceptions of CubeNI and CubeMK in Brazil and Japan.
- Analysis of how different technological familiarization levels impacts the perceived user experience of gestural interfaces.

Chapter 5

Accessible Multi-Scale Gestural Interaction

Multi-scale Interactions refers to the mix of short-scaled interactions, where the user, gesture capturing device (camera) and interaction targets are within 2 meters of each other. It also refers to long-scaled interactions where larger distances are observed either between user-camera or user-targets. When interacting with far away targets, the minimal changes in user movement creates a big variance in the resulting interaction. In this section the author will explain the motivation to use drones in the exploration of multi-scale interactions and the challenges associated with it.

5.1 Introduction

Gesture interfaces are explored mostly in short-scaled interactions [26, 35, 31, 36, 37]. In the short-scaled scenario the distance between user, gesture input device (camera), and targets of interaction does not exceed 2 meters.

The Gesture Development Process presented in Chapter 3 resulted in a gesture for "select" (Figure 3.13). Because of the wide-spread applicability of the "select" system function, the author chose this system function and associated gesture to explore multi-scale gestural interactions.

In recent years, robots have been taking advantage of developments in Hardware and Software Technologies; this continuous improvement prompts the expansion in the potential use of robotics, which leads to crescent visibility of

robots in everyday lives. A specific type of robot that has been gaining popularity is Unmanned Aerial Vehicles (UAVs). Also known as drones, UAVs are flying robots that can be remotely controlled or operated autonomously; they have diverse applications: cinematography, disaster monitoring [56], surveillance, journalism [57], military reconnaissance [58], and others. Given the wide variety of applications and multiple degree of freedom, the author chose to use UAVs as a means to explore multi-scale gestural interactions.

The most basic feature of Human-Drone Interaction is a user or system signaling a drone to change its position [59]. In the traditional form, the user directly controls the flight path using a device like a joystick. With more intelligent systems, how one controls the drone changes. For example, the user can define targets for the drone to reach autonomously [60]. They are thus allowing for applications where the drone could position itself relative to a goal, which in turn would make more complex tasks possible, for example, signaling a drone to grab and drop blocks, stacking them until the drone can build a wall. The scenario just described can be referred to as a Task-Oriented scenario [59].

On Task-Oriented scenarios, drones might need to interact and make decisions based on human input. In order to create user interfaces for human-drone interaction, it is necessary to control the drone, know where the drone is, and provide communication between the drone and other systems [59].

Controlling the drone in interactive systems depends on the available resources and requirements of the task at hand. For example, wearable devices might not be the optimal solution for long jobs where they would run out of batteries [61], or tasks could involve a high number of human participants. Other options could be the traditional Command-Line: every time the drone required a human input in the middle of a task, they could provide it via typing a command. But the downsides to it are clear, seeing as the human might need to interrupt their activities to communicate with the drone. One possible solution to avoid wearable devices, or interrupting human activities to input commands to a system, would be the use of Gesture Input.

Gesture refers to any bodily motion or states, particularly any hand motion or face motion [62]. It is a form of visual communication that, in the case of

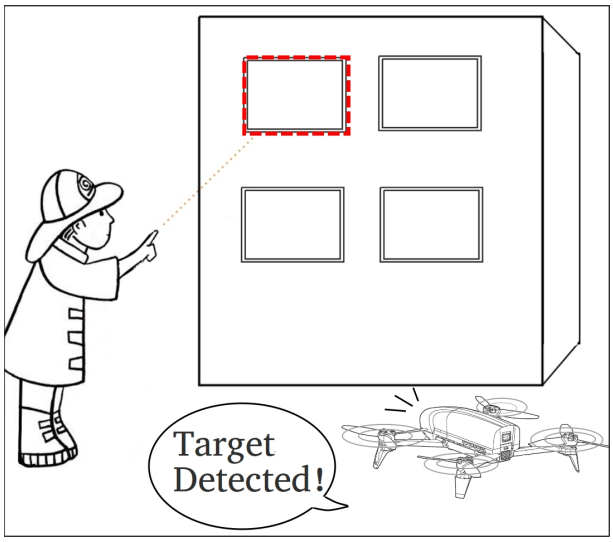


Figure 5.1. Possible scenario for the system presented here

drones, could be especially convenient in cases that computer vision substitutes GPS on determining the drone position, which is the case of indoor environments. It would also benefit situations where computer vision and GPS, or other localization technologies, could cooperate to create a more robust system.

Gestures are also a natural form of human communication. Imagine the following scenario (Figure 5.1): drones are supporting a firefighter unit by entering a building and providing some assessment on the situation inside. In this scenario, a possible approach would be for the drone to wait for a human command, as to which window or entrance it should take to enter the building mentioned above. The firefighter could point to the desired window, and the drone, now with a target entrance, could carry on with some designed task inside, like scanning the room for victims.

There are many possible scenarios where drone systems would benefit from interaction with humans in the course of a task. Thinking of making the research presented in this chapter more accessible, the presented Human-Drone Interactions are explored when relying solely on the drone camera, without external devices, non-dependant on GPS systems, and without the use of external tracking of the drone (for example, external cameras pointing to the drone). The author studied Pointing Gestures as a means to estimate the correct Target Object. The author performed an Elicitation Study with a fire brigade, during

which the firefighters often chose to use pointing gestures and voice commands. The result of the elicitation study confirmed the gesture for "select" (Figure 5.1) resulting from the Gesture Development Process. The author developed a 2D and a 3D gesture interface solution for the Elicitation study scenario. The 3D pointing gesture interface solution obtained a 0.73 F1-Score on average in real-world experiments and 0.58 F1-Score at the maximum distance of 25 meters between drone and building.

5.2 Elicitation Study

In the scenario depicted in Figure 5.1, the hypothesis was that the pointing gesture (from "select" in Figure 3.13) would be used to indicate the target window. In order to be certain that this gesture was the most appropriate for this scenario, the author performed an Elicitation Study in a Fire Brigade in Kobe, Japan. In this Study it was questioned how the fireman would interact with the drone, that is, how the fireman would indicate a window as an entering point to the UAV.

Seven firefighters participated in the Study. The position of the subjects on the fire brigade varied from officer to chief, all participants were male and had no previous experience with UAVs. The author asked the subjects opinion on using UAVs to aid firefighters, if they felt that drones could be useful, and if so, what type of interface would they prefer to control it.

5.2.1 Environment

The first part of the Elicitation Study consisted of the following scenario: a drone hovering next to a building. Each subject stood between the drone and the building (Figure 5.2). Each subject was requested to signal the drone, in whichever way they preferred, to enter a window. Afterward, in the second part of the Study, an interview was conducted with each firefighter. The author collected their opinions on future applications of drones in firefighting, and interfaces for drones control systems.

The author tested the scenario using four different positions between subject and drone (Figure 5.3). In the first setup, the drone was two meters far from the

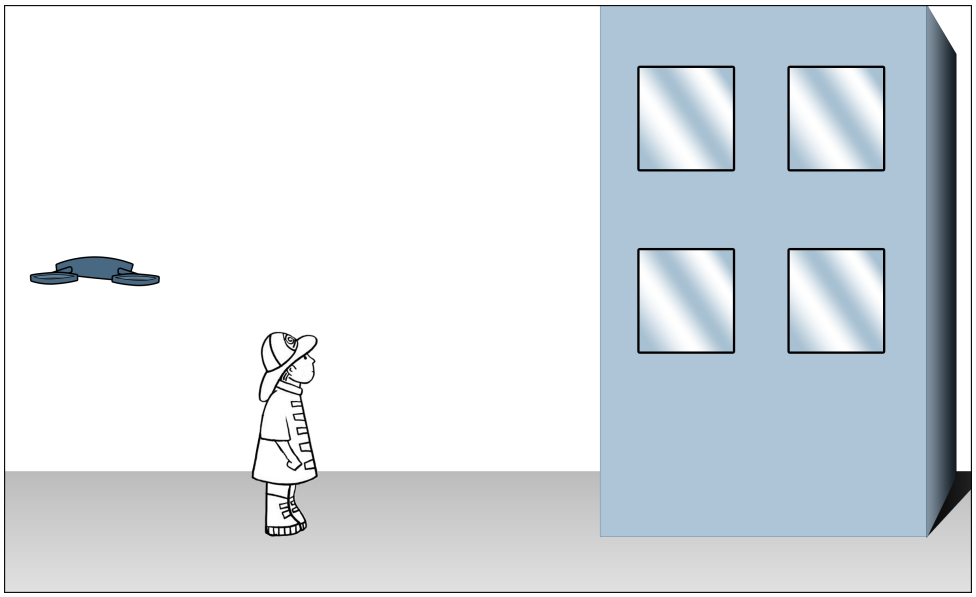


Figure 5.2. Elicitation Study: How to indicate a window to a drone?

subject and hoovered two meters high. In the second setup, the drone was two meters far from the subject and hoovered ten meters high. In the third setup, the drone was ten meters far from the subject and hoovered two meters high, and on the fourth, the drone was ten meters far from the subject and hoovered ten meters high. The distance between a subject and the building was always 40 meters.

5.2.2 Results

All subjects wished for a fully autonomous drone that could quickly arrive in the fire scene and gather useful data for First Responders such as building type, source of fire, type of fire, victims present.

An important observation made was that in dealing with drones, the firefighters would like to be always kept in touch with how the task performed by the drone was evolving, that is, quality and timed feedback to ensure the firefighter the drone was not derailing from the defined task.

When describing the scenario, the word “pointing” was not mentioned. The author didn’t give any suggestions to the Pointing gesture, verbally or otherwise. In the results (Figure 5.4) the firefighters always used Pointing Gesture and

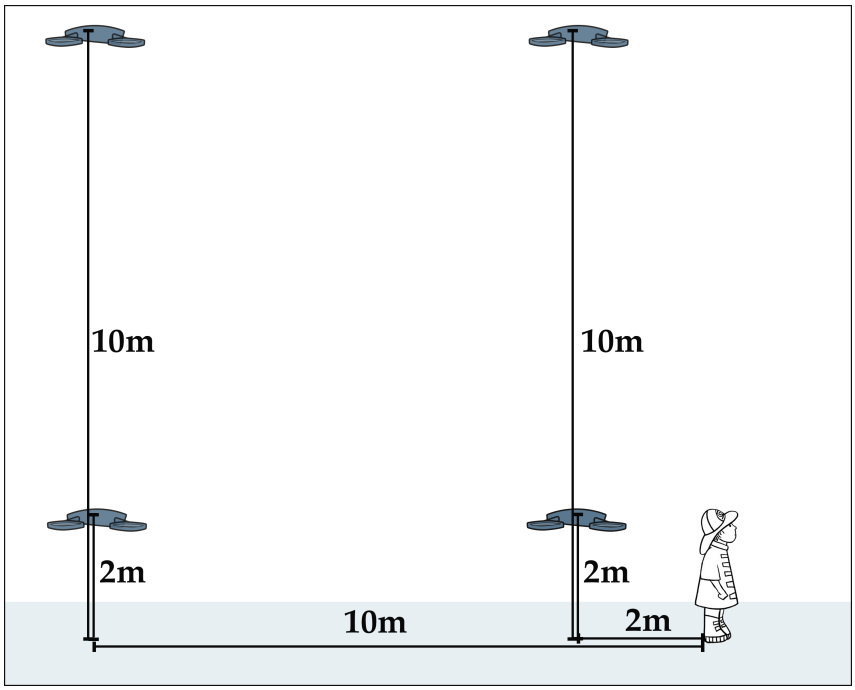


Figure 5.3. The author tested four different positioning between subject and drone

Verbal commands to indicate a window.



Figure 5.4. Experiment: Indicating a window to the drone

Given the results of this Elicitation Study the author became interested in building a gesture interface that used Pointing Gestures to indicate a window

to a drone. Besides the scenario here presented, there are many possible scenarios where drone systems would benefit from interaction with humans in the course of a task. This work aims to provide guidance for natural Human-Drone Interactions when relying solely on the drone camera, without external devices, non-dependant on GPS systems, and without the use of external tracking of the drone (for example, external cameras pointing to the drone). Pointing Gestures are studied as a means to estimate the correct Target Object.

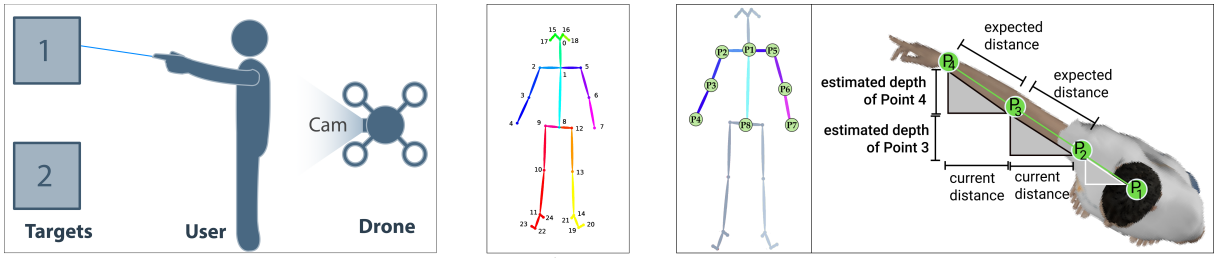
5.3 2D Pointing Gesture Interface Approach

This section explains the first proposed pointing gesture detection method. The author defines a pointing gesture as "a gesture in which the user extends their arm away from their body, towards the desired target object (window) to be selected." In the 2D Pointing Gesture Interface Approach, object detection and pose estimation were combined to decide which object is selected by the user.

There are four essential elements in the system presented here (Figure 5.5(a)): The User, the Targets, the Camera, the Processing Unit. The User is the human who will interact with the Camera, this interaction will be in the form of pointing towards the desired Target. The Targets are types of visual content, for example, an object that the Processing Unit has been trained to recognize by using object detection algorithms like YOLO [63]. The Processing Unit is any computational system that processes the images from the Camera and sends signals to the drone or robot. The Camera is the image feed in the moving robot or drone.

The pointing gesture detection uses OpenPOSE [64] to detect two points on the User's forearms, these two points are used to calculate the equation of a line, extending the line segment until the borders of the image, hereby defined as Extended Pointing Segment. OpenPOSE is a real-time multi-person pose estimation system that detects the human body, hand, face, and foot keypoints on single images. The author used OpenPOSE's BODY_25 (Figure 5.5(b)) model for speed and accuracy.

The Target detection uses YOLOV3-tiny [65], which provides sufficient speed



(a) (b) (c)

Figure 5.5. (a) Scenario Overview There are four essential elements: the user, the targets, camera, and processing unit. The user interacts with the camera in the form of pointing towards the desired target. The targets are types of visual content. For example, this includes a window that the processing unit has been trained to recognize using object detection algorithms. The processing unit is any computational system that processes the images from the camera and sends signals to the UAV. The Camera is the image feed in the moving UAV. (b) Pose Output Format: BODY-25. (c) Left-side: Points used from BODY-25 model. Right-side: Intuition on depth estimation of keypoints. When pointing backwards (transverse and sagittal planes), the projected arm on the picture will be smaller than pointing sideways (frontal plane). Using a right-angle triangle intuition, the calibration data gives us an estimation of the depth value in an arm keypoint.

and accuracy to be run on an integrated development kit in the future. A total of 400 images were used in the training of each target object. The work presented relies on the input of Openpose and YOLO to get the pointing gesture and target objects' Bounding Box points, if the Extended Pointing Segment and the Bounding Box of a Target Object are overlapping, this means that that target object was selected. The drone now can interact with this Target, for example getting closer to it.

From this model, there can be obtained two points on a forearm, these two points are used to calculate the equation of a line, making possible to extend the forearm's line segment until the borders of the image, hereby referenced as Extended Pointing Segment.

The 2D approach starts by acquiring an RGB image from a UAV. Each monocular image has dimensions represented by a width w and height h. Openpose Body25 model uses non-parametric representations called Part Affinity Fields to regress joint positions and body segment connections between the joints. The BODY_25 model provides P_n points where $n \in \mathbb{N}$, $0 \leq n \leq 24$. The author only uses eight points: P_1 until P_8 (Figure 5.5(c), left-side). Each point $P_n \in \mathbb{R}^{w \times h}$ has 2D coordinates P_n^x and P_n^y, the pointing gesture is defined by the arm keypoints: P_3, P_4 and P_6, P_7. Notice that P_5, P_6 and P_7 are analogous to P_2, P_3 and P_4, respectively. Two points from the same arm are used to calculate a 2D line equation, which can take the form

$$f(x) = \frac{(P_3^y - P_4^y)}{(P_3^x - P_4^x)} \times (x - P_3^x) + P_3^y \tag{5.1}$$

for points P_3, P_4. For each arm, a line segment is defined starting from an elbow point like P_3 in the direction of a hand point like P_4 up to the border of the image, hereby referenced as Extended Pointing Segment. The borders of the image can be defined as $f(y) = w$, $f(y) = 0$, $f(x) = h$,$f(x) = 0$, for every $(x, y) \in \mathbb{R}^{w \times h}$. The choice of which border to use is guided by the direction of the pointing gesture.

The author uses YOLO [65] to detect objects. It produces bounding boxes for the objects it was trained to recognize. The bounding boxes are simply two

points $B_{n1} \in \mathbb{R}^{w \times h}$ and $B_{n2} \in \mathbb{R}^{w \times h}$ used to define a rectangle that contains the object on the image.

The Target Object detection relies on YOLO, which same as OpenPOSE, is used inside ROS environment. For each detected object it creates a Bounding Box. The Bounding Box consists in two points, the top left and top right points of an object. In order to Select an Target Object there is a check to see if the Extended Pointing Segment and the Bounding Box are overlapping. The details of how this check was performed will be described below.

A Bounding Box has four line segments that together form a rectangle. The Pointing Gesture is a line segment that originates in the User's elbow, passing by User's wrist and is extended until the margins of the image. The check performed is whether or not the Pointing Gesture's line segment (Extended Pointing Segment) crosses with any of the Bounding Box's four line segments. To better visualize what was said, Figure 5.6 depicts the Extended Pointing Segment as the line segment from point P until point Q, and the Bounding Box as a rectangle with vertices ABCD.

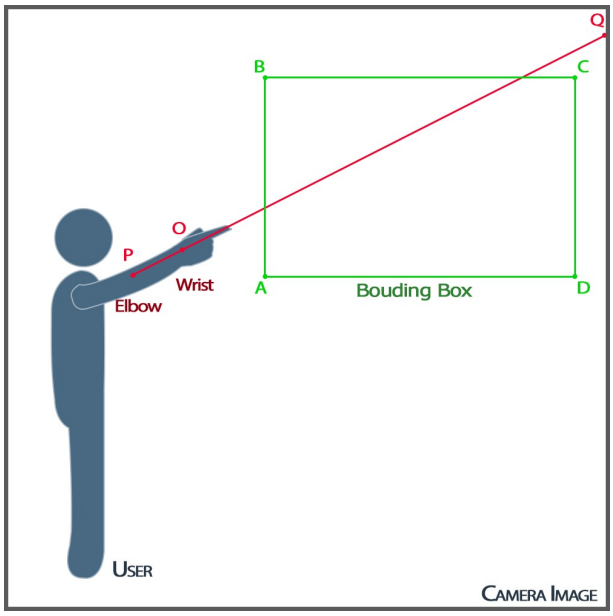


Figure 5.6. Target Object Estimation using Pointing Gesture

In order for the Object delimited by the ABCD bounding box to be selected, the Extended Pointing Segment PQ has to cross either AB, BC, CD or AD.

There is an approach to this problem that uses Vector Cross Products [66]. Define the 2-dimensional vector cross product $\vec{v} \times \vec{w}$ to be $\vec{v}_x\vec{w}_y - \vec{v}_y\vec{w}_x$. Consider P and Q to be the points that stand at the end of vectors $\vec{p}$ and $\vec{p}+\vec{r}$, respectively. Also, consider A and B to be the points that are located in the end of vectors $\vec{a}$ and $\vec{a}+\vec{s}$, as shown in Figure 5.7.

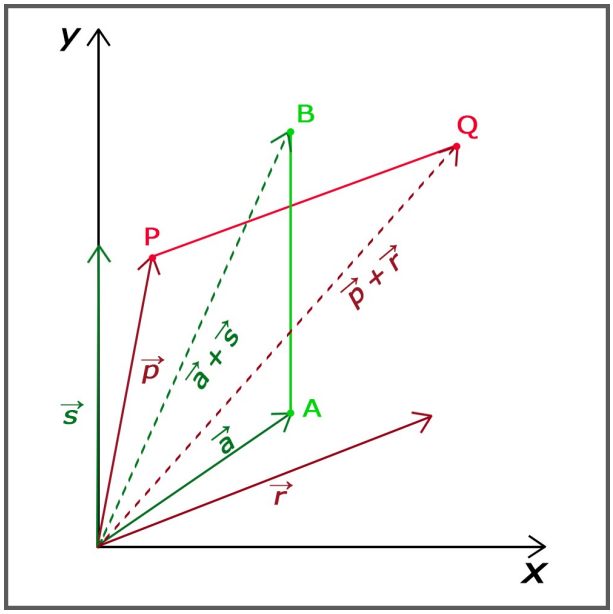


Figure 5.7. Vector Cross Products approach to verify if two line segments are crossing each other

Then any point on the line segment PQ can be represented as $\vec{p}+t\vec{r}$ (for a scalar parameter t) and any point on the line segment AB as $\vec{a}+u\vec{s}$ (for a scalar parameter u). The two lines intersect if there is a t and u such that $\vec{p}+t\vec{r} = \vec{a}+u\vec{s}$, as shown in Figure 5.8.

In order to isolate t we cross both sides with $\vec{s}$ (5.2), and to isolate u we cross with both sides with $\vec{r}$ (5.3).

$$(\vec{p}+t\vec{r}) \times \vec{s} = (\vec{a}+u\vec{s}) \times \vec{s} \tag{5.2}$$

$$(\vec{p}+t\vec{r}) \times \vec{r} = (\vec{a}+u\vec{s}) \times \vec{r} \tag{5.3}$$

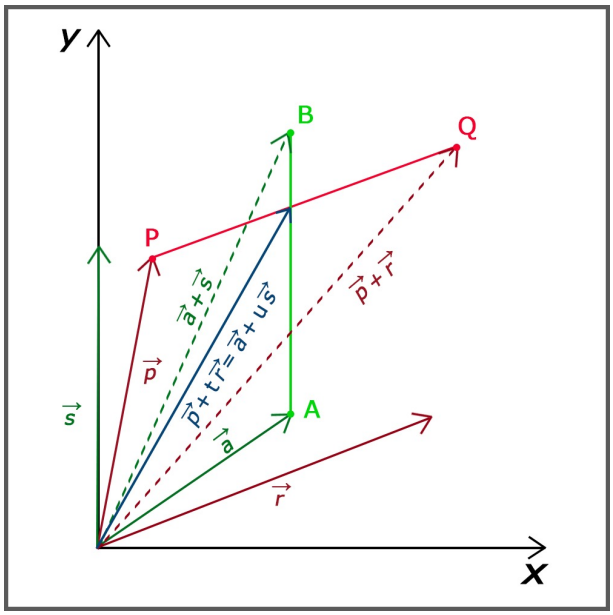


Figure 5.8. If $\vec{p}+t\vec{r}=\vec{a}+u\vec{s}$ then the two lines intersect.

Since $\vec{r}\times\vec{r}=0$ and $\vec{s}\times\vec{s}=0$, this means the following:

$$(\vec{p}+t\vec{r})\times\vec{s}=(\vec{a}+u\vec{s})\times\vec{s}$$
$$(\vec{p}\times\vec{s})+t(\vec{r}\times\vec{s})=(\vec{a}\times\vec{s})+u(\vec{s}\times\vec{s})$$

$$t=\frac{(\vec{a}-\vec{p})\times\vec{s}}{\vec{r}\times\vec{s}} \tag{5.4}$$

Similarly, for u:

$$(\vec{p}+t\vec{r})\times\vec{r}=(\vec{a}+u\vec{s})\times\vec{r}$$
$$(\vec{p}\times\vec{r})+t(\vec{r}\times\vec{r})=(\vec{a}\times\vec{r})+u(\vec{s}\times\vec{r})$$
$$u=\frac{(\vec{p}-\vec{a})\times\vec{r}}{\vec{s}\times\vec{r}}$$

Given that $\vec{s}\times\vec{r}=-\vec{r}\times\vec{s}$, it is convenient to rewrite u as follows:

$$u=\frac{(\vec{a}-\vec{p})\times\vec{r}}{\vec{r}\times\vec{s}} \tag{5.5}$$

Given the above, there are four cases to consider. First, in the case that $\vec{r}\times\vec{s}=0$ and $(\vec{a}-\vec{p})\times\vec{r}=0$, then the two lines (PQ and AB) are collinear. In

this case, it is possible to express the endpoints of the AB segment ($\vec{a}$ and $\vec{a}+\vec{s}$) in terms of the equation of the PQ line segment ($\vec{p}+t\vec{r}$), as shown in Figure 5.9.

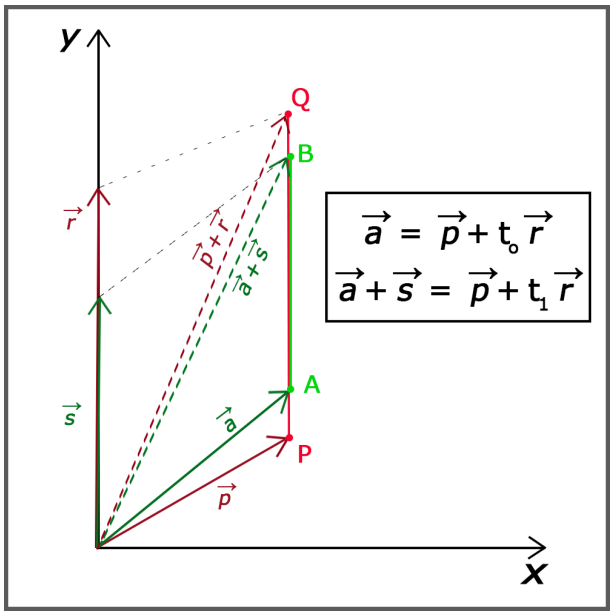


Figure 5.9. Case when the Extended Pointing Segment PQ and Bounding Box segment AB are collinear.

Developing the equation for t_0 (5.6):

$$\vec{p}+t_0\vec{r} = \vec{a}$$
$$\vec{p}+t_0\vec{r} = \frac{\vec{a}\cdot\vec{r}}{\vec{r}}$$
$$\vec{r}\cdot(\vec{p}+t_0\vec{r}) = \vec{a}\cdot\vec{r}$$
$$t_0\vec{r}\cdot\vec{r} = (\vec{a}-\vec{p})\cdot\vec{r}$$

$$t_0 = \frac{(\vec{a}-\vec{p})\cdot\vec{r}}{\vec{r}\cdot\vec{r}} \tag{5.6}$$

Developing the equation for t_1 (5.7):

$$\vec{p} + t_1\vec{r} = \vec{a} + \vec{s}$$
$$\vec{p} + t_1\vec{r} = \frac{(\vec{a} + \vec{s}) \cdot \vec{r}}{\vec{r}}$$
$$\vec{r} \cdot (\vec{p} + t_1\vec{r}) = (\vec{a} + \vec{s}) \cdot \vec{r}$$
$$t_1\vec{r} \cdot \vec{r} = (\vec{a} + \vec{s} - \vec{p}) \cdot \vec{r}$$

$$t_1 = \frac{(\vec{a} + \vec{s} - \vec{p}) \cdot \vec{r}}{\vec{r} \cdot \vec{r}} = t_0 + \vec{s} \cdot \frac{\vec{r}}{\vec{r} \cdot \vec{r}} \tag{5.7}$$

It can be observed that if the interval between t_0 and t_1 intersects the interval $[0,1]$ then the line segments AB and PQ are collinear and overlapping, otherwise they are collinear and disjoint. In the case that $\vec{s}$ and $\vec{r}$ are pointing in opposite directions, the interval to be checked are $[t_1, t_0]$ rather than $[t_0, t_1]$. The Target Object would not be considered selected if they were disjoint.

The second case of the four possible cases would happen if $\vec{r} \times \vec{s} = 0$ and $(\vec{a} - \vec{p}) \times \vec{r} \neq 0$, then the two line segments AB and PQ are parallel and non-intersecting (Figure 5.10), therefore the Target Object wouldn't be selected.

The third case happens when $\vec{r} \times \vec{s} \neq 0$ and $0 \leq t \leq 1$ and $0 \leq u \leq 1$, then the two line segments AB and PQ would meet at the point $\vec{p} + t\vec{r} = \vec{q} + u\vec{s}$. In this situation, regarding just the line segment AB of the Bounding Box ABCD, the Target Object delimited by it would be considered selected. Therefore, for a Target Object given by a Bounding Box ABCD to be considered as selected, the Extended Pointing Segment PQ only have to be crossing with one of the four segments of the ABCD Bounding Box.

The fourth and last case would happen in a three-dimensional environment, when the two line segments are not parallel but do not intersect. The solution for this 3D scenario is addressed in the next section, 3D Pointing Gesture Interface Approach.

With the Extended Pointing Segment and the Bounding Box are overlapping, a Target Object is now defined and selected. In the case that the Extended Point-

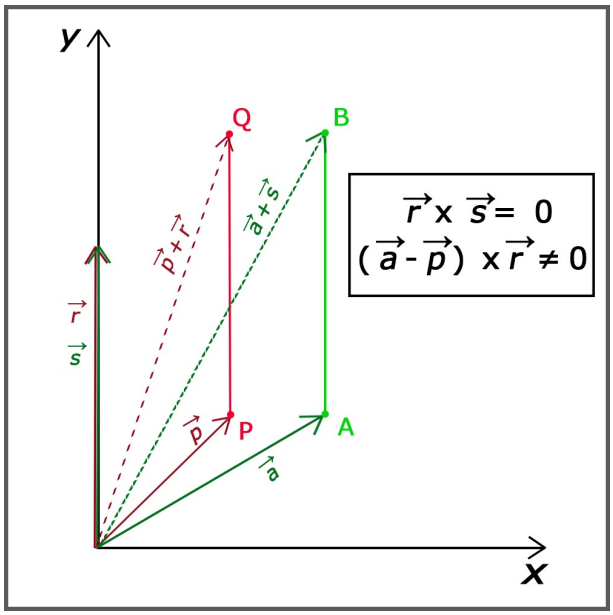


Figure 5.10. Case when the Extended Pointing Segment PQ and Bounding Box segment AB are parallel and non-intersecting.

ing Segment crosses more than one bounding box, the 2D approaches chooses the bounding box that has the center closest to the Extended Pointing Segment. A flowchart of the whole system can be seen on Figure 5.11.

The drone uses a PID controller to reach the Target Object, using only 2D coordinates from the UAV's monocular camera. In order to do that, the center of the image and the center and area of the Target Object is taken into account. On the drone image feed, if the center of the Target Object is to the right of the middle point of the image, than the drone should go right. Likewise for the left, up and down sides. Here it should be mentioned that the bebop 2 camera stabilization of the image influences the movements going up and down, because the motion effect on those directions are masked by it. Therefore the disabling of this feature is suggested. Lastly, to calculate the approximation to the Target Object, its area is taken into account.

This system works in outdoor and indoor environments. If the user points in the general location of a target object (e.g., window), the system will select it, and there is no need to point directly towards the object. But there are shortcomings: a line crossing multiple windows is ambiguous, if there are multiple

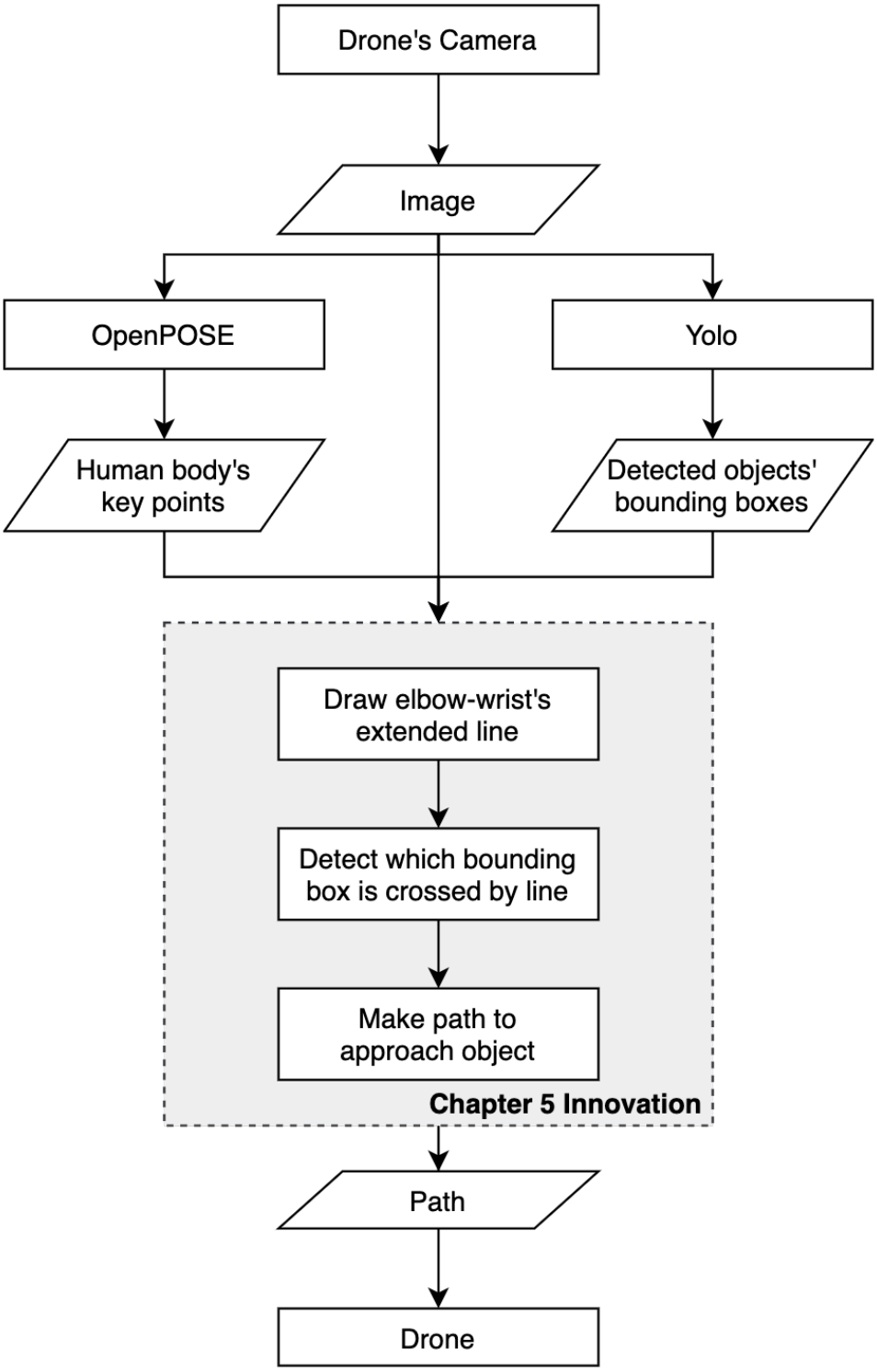


Figure 5.11. Flowchart of the 2D Approach

windows around the correct one, it is not guaranteed that the pointing line will cross the correct one closest to the center (see Figure 5.20(a)), and it is not guar-

anteed that the user will point to the center of the window. Therefore the lack of depth information to make a more informed decision, limits this system. In order to overcome that limitation, the author proposes the 3D pointing gesture interface approach.

5.4 3D Pointing Gesture Interface Approach

To estimate the correct pointing area, or pointed object (window), 3D information is required. We can capture the 3D information of a scene in the format of point clouds. Point clouds can be obtained from diverse sources, the author chose a monocular SLAM, named ORB-SLAM, that is relatively low cost and that produced a sparse point cloud with enough information for our purposes: Obtaining the distances from drone to user and to target.

ORB-SLAM [67] is a feature-based monocular system that operates in real-time in indoor and outdoor environments. The author was interested in the sparse point cloud it provides, and modified its original code to output the point cloud of the feature points detected on the current image frame. By combining object detection and pose estimation algorithms, in the author's case YOLO and openPOSE, it is possible to estimate the depth of objects of interest on the current RGB image. This monocular SLAM provides an unscaled point cloud. To scale it, the author performs a one-time calibration step as explained below. The author also provides a general model, using average user values, in case this calibration process is not feasible. With depth estimated, it is possible to calculate the pointing gesture direction up to a specific point in the monocular image. A flowchart of this approach is shown in Figure 5.12.

This approach starts by acquiring an RGB image from a UAV. Each image frame has a width w and height h. The X axis represents the width, the Y axis the height, with origin on the upper left of the image. The author integrates an estimation of the Z axis pointing towards the inside of the image.

The BODY_25 model (Figure 5.5(b)) provides 25 keypoints $P_n \in \mathbb{R}^{w \times h}$ where $n \in \mathbb{N}$, $0 \leq n \leq 24$. Each point has 2D coordinates P_{nx} and P_{ny}. The author first performs a calibration step; this is simply to create a file to store the values of all

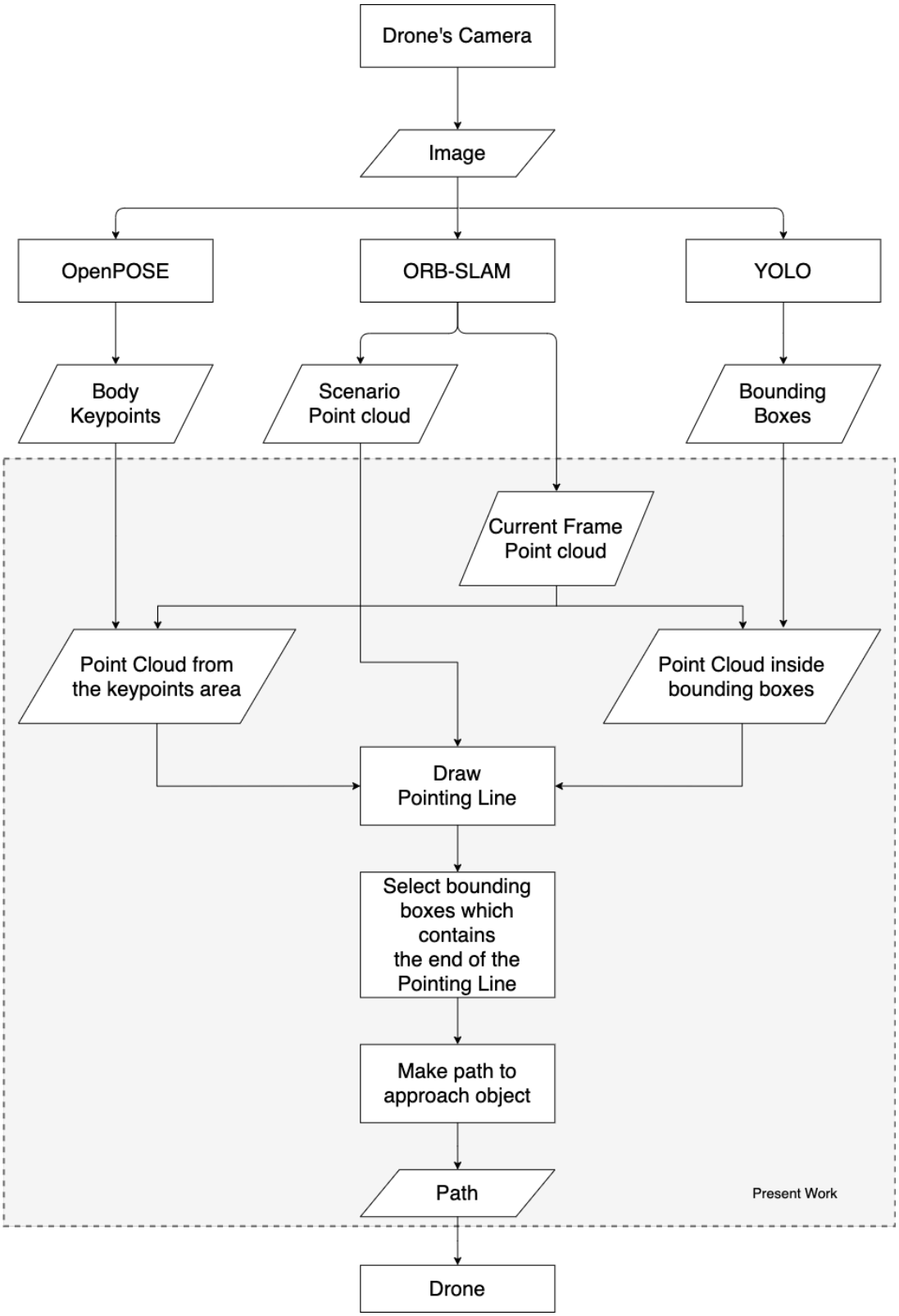


Figure 5.12. 3D pointing gesture interface application flowchart. ORB-SLAM integrated with Object Detection and Pose Estimation towards pointing gesture interaction.

the user's keypoints at a fixed distance D_f to the UAV. The author also stores the real measurements of the user's height and torso size. The three planes of body motion are the sagittal, frontal, and transverse. In the calibration step, the user is standing with open arms at the side of the body, frontal plane. The calibration step takes about a minute and is a one-time step for each user.

5.4.1 Calibration

The calibration step stores the size of the user's torso, in pixels, given by the euclidean distance between points P_1 and P_8:

$$d_f(P_1,P_8) = \sqrt{\left(P_1^x - P_8^x\right)^2 + (P_1^y - P_8^y)^2}, \text{ at distance } D_f \tag{5.8}$$

at a fixed distance D_f to the UAV. Therefore D_f will refer to the distance to the user in calibration time. In a real situation, the user moves away or towards the UAV, therefore the current distance D_c from the user to the UAV's camera (UAV) changes. To estimate D_c we use the current distance d_{c18} between points P_1 and P_8, and the inverse linear relationship between size and distance:

$$D_c = \frac{D_f \times d_{f18}}{d_{c18}} \tag{5.9}$$

$$d_{c18}(P_1,P_8) = \sqrt{\left(P_1^x - P_8^x\right)^2 + (P_1^y - P_8^y)^2}, \quad \text{at distance } D_c \tag{5.10}$$

where d_{c18} is the current euclidean distance between points P_1 and P_8, and d_{f18} is the fixed distance between points P_1 and P_8. That is, d_{f18} is the distance between points P_1 and P_8 at calibration time.

When moving, if the user's arm stays in the frontal plane, the distances in between his arm's keypoints is decreased by $\frac{D_f}{D_c}$. However, when the user points to something behind himself or herself, moving the arm in the transverse and sagittal planes, the projection of the arm on the image plane will be smaller than expected. The author assumes that the depth P_1 will be the current distance D_c to the UAV. To express that depth in pixel units, we use the proportion from the

torso pixel size in calibration time d_{f18} and real size tx (metric units):

$$P_{1z} = \frac{D_c \times d_{f18}}{tx} \tag{5.11}$$

Using a right-angle triangle the author calculates the estimated depth on an arm keypoint. This intuition is better visualized on Figure 5.5(c).

5.4.2 Depth Estimation

The estimated depth P_4^z for the hand considers the preceding keypoints' depths P_3^z, P_2^z and P_1^z. The estimated depth P_2^z for P_2 uses the current distance d_{c12} from P_1 to P_2, the saved distance d_{f12} from P_1 to P_2 at calibration time, the current distance d_{c18} from P_1 to P_8, and the saved distance d_{f18} from P_1 to P_8 at calibration time:

$$P_2^z = P_1^z + \sqrt{\left(\frac{d_{c18} \times d_{f12}}{d_{f18}}\right)^2 - (d_{c12})^2} \tag{5.12}$$

$$P_3^z = P_2^z + \sqrt{\left(\frac{d_{c18} \times d_{f23}}{d_{f18}}\right)^2 - (d_{c23})^2} \tag{5.13}$$

$$P_4^z = P_3^z + \sqrt{\left(\frac{d_{c18} \times d_{f34}}{d_{f18}}\right)^2 - (d_{c34})^2} \tag{5.14}$$

knowing P_2^z and P_3^z, the author can estimate P_4^z. Analogously, the author obtains P_5^z, P_6^z and P_7^z. We now have (x,y,z) coordinates, in pixels, for every point on the arms. At this point, we just need the depth of the building to finish the pointing line, from the user to the building direction.

5.4.3 3D Target Estimation

Knowing the estimated user distance in the current frame, the author can estimate the distance to the building with ORB-SLAM. It is an ORB-feature based SLAM that provides a point cloud in the form of map points and its own definition of keyframes. Map points are the structure for the 3D reconstruction of

the scene. Each map point corresponds to a textured planar patch in the world whose position has been triangulated from different views [68]. Points correspond to ORB features in the images, so map points are triangulation of FAST corners. The usual output is the map points for the whole world. The author modified the original code to also output all map points corresponding to the ORB features in just the current image. OpenPOSE gives the position of the user on the current image, and YOLO, the windows. Because it is possible to get the map points associated with each one, it gives the author the unscaled relationship between the user distance to the UAV and the building (windows). The author uses this data and the estimated user distance D_c to estimate the building distance B_c.

With the building distance B_c known, the product $b_c = \frac{B_c \times d_{f18}}{tx}$ will be the conversion from the distance B_c, in metric system units, to the author's system Z axis, in pixel units. Given two keypoints from the arm, it is possible to calculate the 3D line equation, and find (x_c, y_c) values given b_c as a z value, so for the keypoints P_3 and P_4:

$$\alpha\left(\frac{b_c - P_3^z}{P_4^z - P_3^z}\right) = \frac{\boldsymbol{x_c} - P_3^x}{P_4^x - P_3^x} = \frac{\boldsymbol{y_c} - P_3^y}{P_4^y - P_3^y} \tag{5.15}$$

α is needed because the 3D line equation belongs in an orthogonal projection, and this system deals with perspective projection. This wouldn't be a problem in short-scaled interactions but it is a big challenge in monocular-based multi-scaled interactions. The calculation of α is part of the innovation presented in this chapter, as it was obtained after numerous tests in simulation. Therefore, α is a scaling factor associated with B_c in the following way:

$$FOV_x = 2 \times arctan\left(\frac{w}{2 \times F_x}\right) \tag{5.16}$$

$$a = 2 \times D_c \times tan\left(\frac{FOV_x}{2}\right) \tag{5.17}$$

$$b = 2 \times tan\left(\frac{FOV_x}{2}\right) \tag{5.18}$$

Finally, α can be obtained by

$$\begin{aligned} \alpha &= \frac{a}{(B_c - D_c) \times b + a} \\ \alpha &= \frac{D_c}{B_c} \end{aligned} \tag{5.19}$$

in equation 5.16, the Field-of-View of the Image is calculated with respect to the Focal Length F_x of the camera's Intrinsic Matrix. In equation 5.17 a is the maximum width, in metric units, in the Field-of-View at distance D_f of the camera. In equation 5.18, b is the change in the maximum width at every change in the distance to the camera. Lastly, α can be defined as the proportion between the distance to the camera (D_c) and the building distance (B_c), as seen in Figure 5.13.

The result is a target point $T_P \in \mathbb{R}^{w \times h}$ where

$$T_P = (x_c, y_c, b_c) \tag{5.20}$$

If T_P is located inside a window's bounding box, that window is selected. Each bounding box is defined by two points $B_{n1} \in \mathbb{R}^{w \times h}$ and $B_{n2} \in \mathbb{R}^{w \times h}$, a rectangle that contains the object on the image.

In the following section, the author describes the experiments performed with the proposed systems. The author tested both systems in a simulation environment and real-world scenario.

5.5 Experiments and Results

The author ran experiments in a simulated and real-world environment on both applications: 2D pointing gesture interface and 3D pointing gesture interface. The results are reported in this section.

5.5.1 Simulation Experiment

The author used Gazebo [69] 7.0 to create a simulated environment with a building model, a human model, and a UAV model. The author modified all models

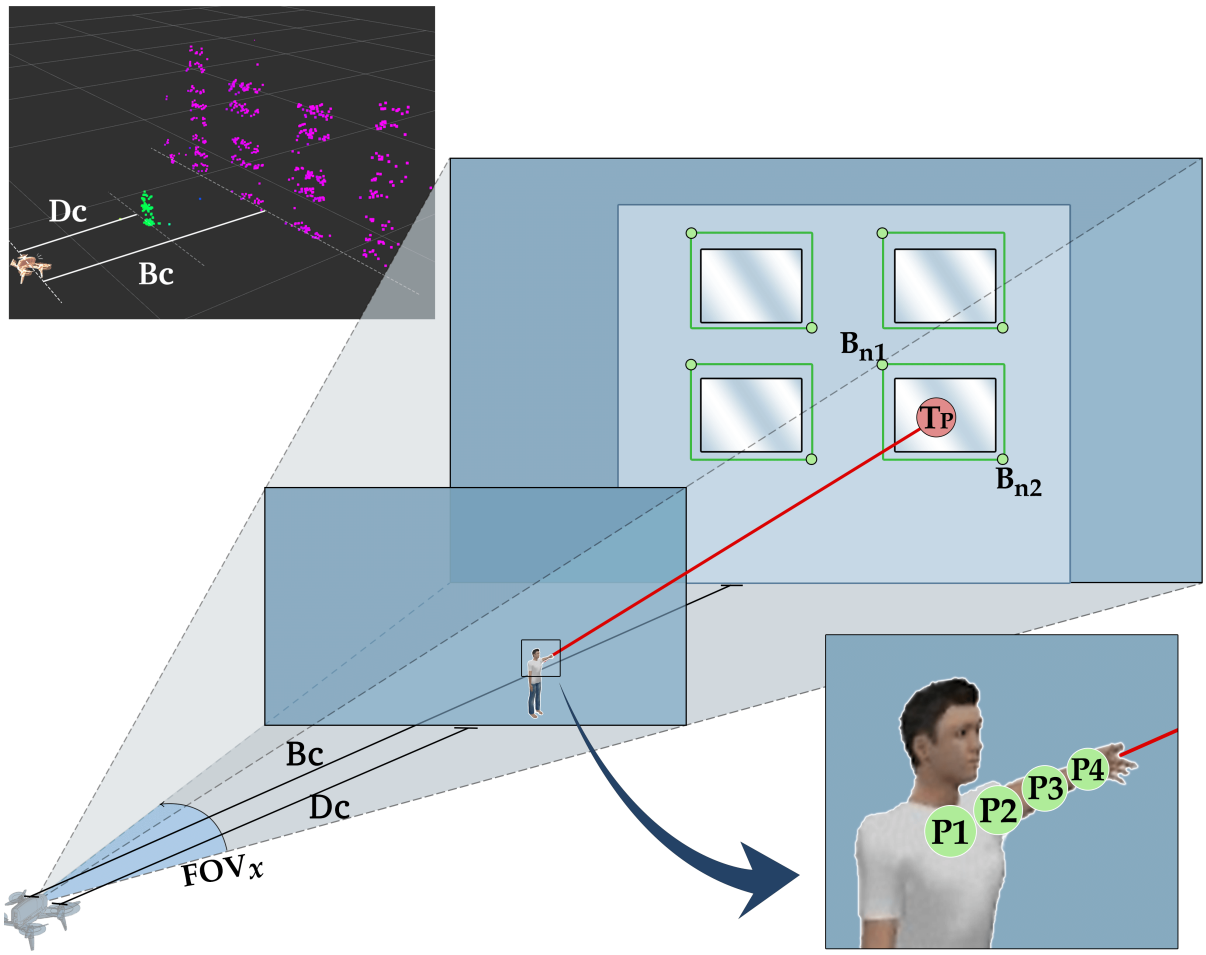


Figure 5.13. Slice of perspective projection at current user distance D_c and current building distance B_c. The 3D pointing gesture system outputs a target point T_P that, if inside a detected window area, selects that window.

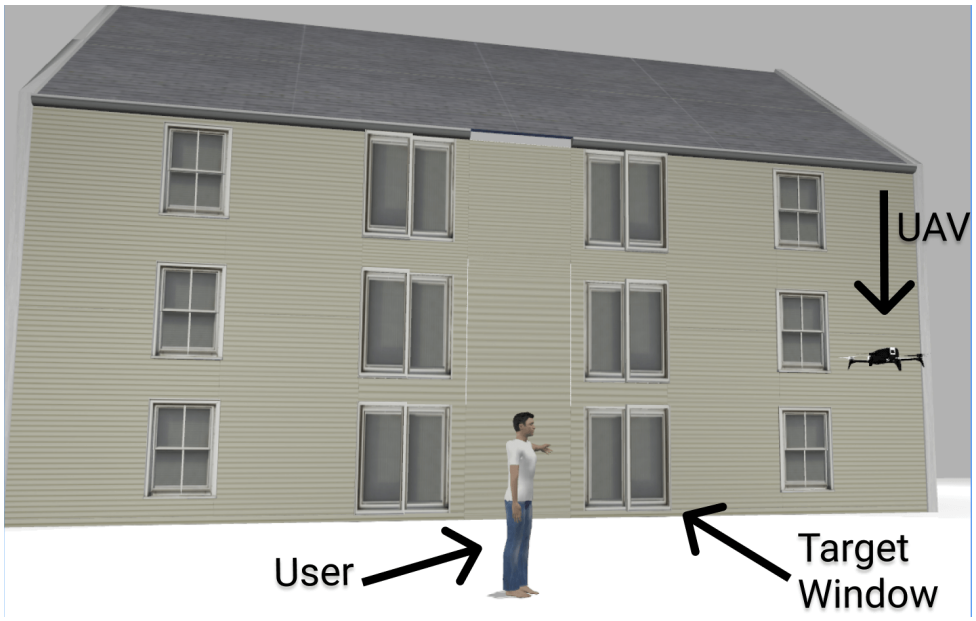


Figure 5.14. Simulation Environment The author used Gazebo 7.0, all models used were modified from Gazebo's originals.

from Gazebo's originals (Figure 5.14). The experiment had eight different positions for the human (Figure 5.15). In some Experiments, the human model executed three fixed pointing angles (Figure 5.16), for each one of the eight positions. In other experiments, the human model would point to four windows for each position. The UAV stood at a fixed distance from the building, and a fixed height of 2.5 meters from the ground.

For each pointing gesture in the Experiments, there is a ground truth associated. This ground truth is either a specific window or that no windows were selected in that case (e.g., pointing to the wall). The author considered a true positive when the system correctly identifies the specific window. True negative means that the system correctly identifies when the pointing gesture is aimed at a not-window object or something outside the UAV's view. False positive happens when the wrong window is identified. False negative means that there is a window being pointed at but the system wrongly concluded that the user is not pointing to any windows. The results from the technologies used here (OpenPOSE, YOLO, ORB-SLAM) are non-deterministic, so when deciding the outcome of the pointing gesture, the author consider a four seconds sequence of

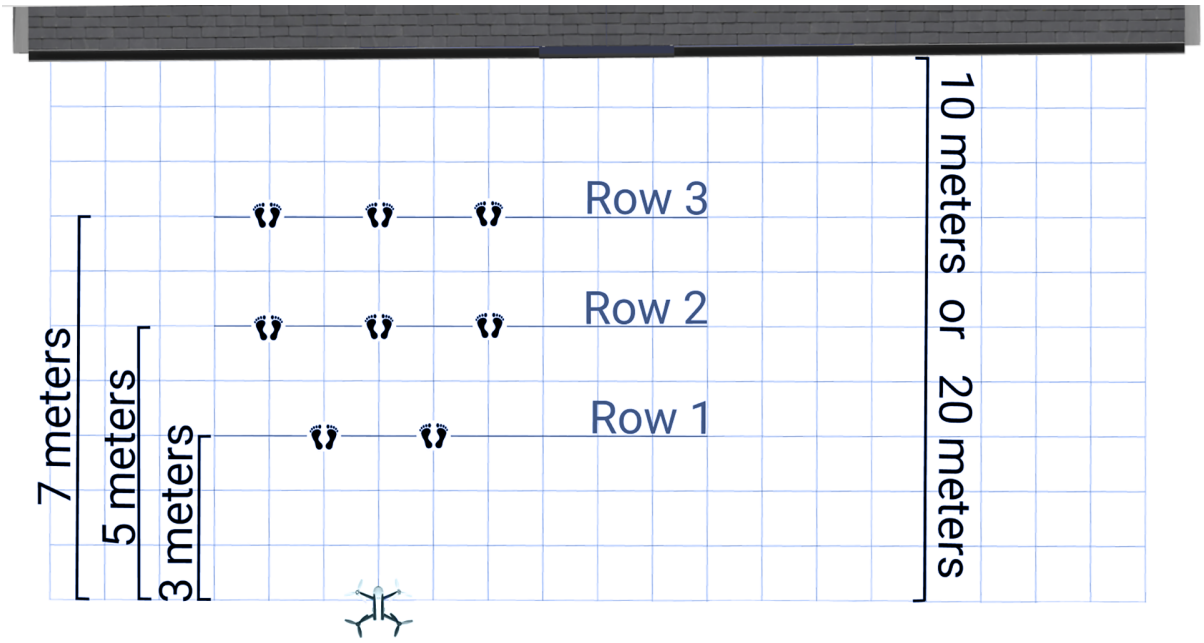


Figure 5.15. User's positioning on the experiments. The author tested the human model on eight different positions, for each Experiment.

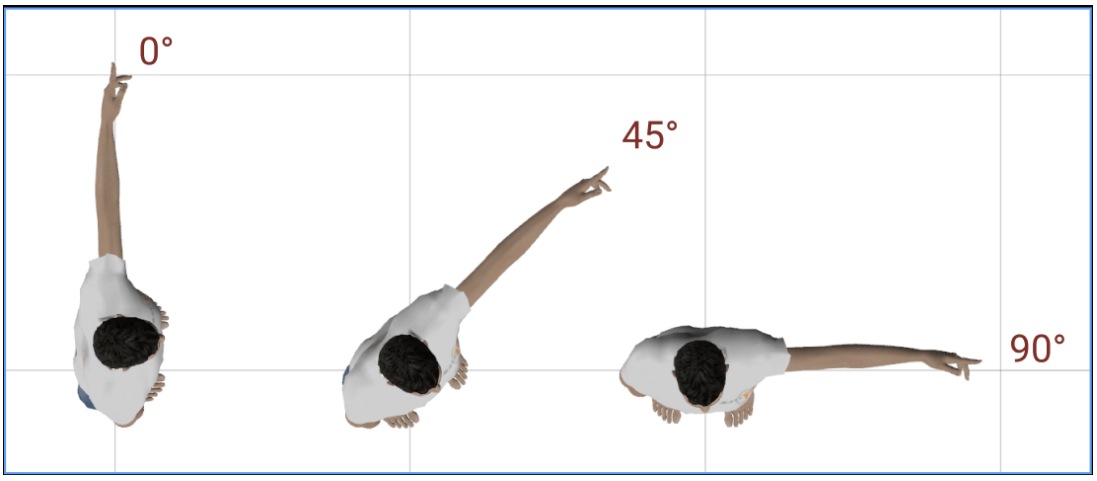


Figure 5.16. Fixed Angles Used. The author experimented with fixed angles, that meant that in some positions the human model would be pointing to the wall, or to outside the UAV's view.

frames to identify the Mode and make a decision.

2D Pointing Gesture Interface Approach Result

The author tested the 2D approach using the eight positions shown in Figure 5.15. The building stood at 10 meters from the UAV. This Experiment is labeled "Exp0". For each position, the human model tested all three fixed angles, as shown in Figure 5.16. This approach could not identify when the human model pointed to not-windows objects; it always tries to select a window. If at least one window bounding box crosses the pointing line, this approach will select it even if the user is pointing at the wall. Results showed 79% of false positives and 21% of true positives. The 2D approach did not produce any false negatives or true negatives. The resulting accuracy was 20%, precision 20%, recall 100%, and F1 Score of 0.34. The Matthew Correlation Coefficient could not be calculated due to zeroed false negatives and true negatives. The results are summarised on Table 5.1.

Table 5.1

Summary of Experiments on Simulation Environment

	2D Exp0	3D Exp1	3D Exp2	3D Exp3	3D Row1	3D Row2	3D Row3	3D Exp123
UAV-Building Distance	10m	10m	20m	10m	N/A	N/A	N/A	N/A
UAV-User Distance	N/A	N/A	N/A	N/A	3m	5m	7m	N/A
MCC	N/A	0.639	0.683	N/A	0.891	0.631	0.500	**0.701**
F1 Score	**0.344**	0.75	0.75	0.933	0.952	0.814	0.740	**0.857**
Accuracy	0.20	0.83	0.83	0.875	0.94	0.81	0.74	0.84
Precision	0.20	0.66	1	1	1	0.84	0.83	0.91
Recall	1	0.85	0.66	0.87	0.90	0.78	0.66	0.80
True Positives	20.8%	25%	25%	87.5%	55.5%	40.7%	37%	45.8%
True Negatives	0%	58.3%	58%	0%	38.8%	40.7%	37%	38.8%
False Positives	79.1%	12.5%	0%	0%	0%	7%	7%	4.1%
False Negatives	0%	4%	16.6%	12.5%	5%	11.1%	18.5%	11.1%

3D Pointing Gesture Interface Approach Result

The author tested the 3D pointing gesture interface approach in three Experiments. The first Experiment was labeled "Exp1". Exp1 used the eight user positioning shown in Figure 5.15, with fixed pointing angles (Figure 5.16). The

building stood at 10 meters of distance from the UAV. Results showed approximately 25% of true positives, 58% of true negatives, 12% of false positives, and 4% of false negatives. The resulting accuracy was 83%, precision was 66%, recall was 85%, F1 Score was 0.75, and Matthew Correlation Coefficient was 0.63. This result is comparably better than the previous one, so the author decided to perform other experiments with the 3D pointing gesture interface approach.

The second Experiment was labeled "Exp2". In Exp2, the author used eight positions with fixed angles, but the building stood 20 meters away from the UAV. The author wanted to see how it behaved with targets (windows) further away. Results showed approximately 58% true positives, 25% true negatives, 16% false positives, and 0% of false negatives. The resulting accuracy was 83%, precision was 100%, recall was 66%, F1 Score was 0.75, and the Matthew Correlation Coefficient was 0.68. The results were slightly better, given the increased distance to the building, but still similar. The author decided to perform another experiment to test pointing at the windows with varied angles instead of using fixed angles.

The third Experiment was labeled "Exp3". For each one of the eight positions in Figure 5.15, the human model would point to all four bottom windows of the building, using angles that best suited each window. Results showed approximately 87% true positives, 0% true negatives, 0% false positives, and 12% false negatives. The resulting accuracy was 87%, precision was 100%, recall was 87%, and F1 score was 0.93. The Matthew Correlation Coefficient was not applicable due to the lack of negatives in the ground truth. The results were slightly better than the previous experiments. The author noticed that when the human model stood at Row 3 (Figure 5.15), OpenPOSE would output less than reliable poses. The author decided to analyze the data from the previous three Experiments by Row. The results (Table 5.1) show that indeed the first row had better results than row 2, which had better results than row 3.

When considering all three experiments, results showed approximately 45% of true positives, 38% of true negatives, 4% of false positives, and 11% of false negatives. The resulting accuracy was 84%, precision 91%, recall 80%, F1

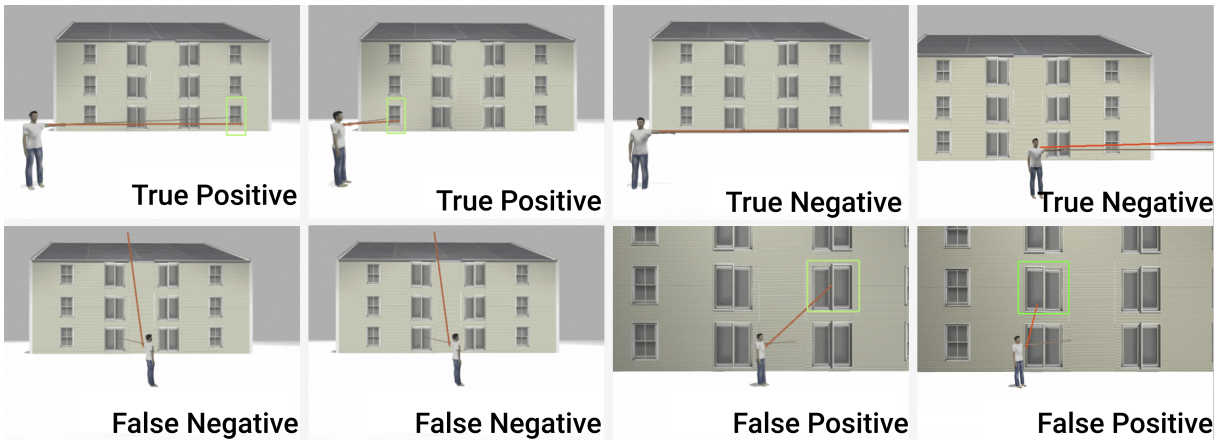


Figure 5.17. Sample of Simulation Experiments' Results

Score of 0.85, and Matthew Correlation Coefficient of 0.70. A sample of successful and failed cases can be seen in Figure 5.17.

5.5.2 Real-World Experiment

An experiment was conducted at the Kobe Fire Academy, Japan. The experiment's purpose was the identification of the user's pointed window by the UAV. The author used the Parrot Bebop 2 [70] for the experiments; it is a quadcopter that weighs around 500 grams, offers 25 minutes of autonomous flight time, can film in 1080p full HD with its wide-angle 14-megapixel lens. But for the sake of performance, the image size was limited to 856 x 480 pixels. The author used an available SDK by the name of bebop_autonomy for the ROS-supported system.

The Bebop 2 interfaced via wifi with a laptop. The author used an Alienware 13 R3, with an NVIDIA GTX 1060 graphics card, 16GB RAM, and an i7-7700hq processing unit. Other than the Bebop 2 and Laptop, no other electronic devices were used. They interfaced using the Robot Operating System (ROS) [71].

The author asked permission from firefighters and received permission from Kobe Fire Academy to perform the experiment. Regarding the use of the aerial robot, there are legal restrictions. This research has got permission from the Ministry of Land, Infrastructure, Transport and Tourism, and Osaka Aviation

Bureau under license number: P180900923 for performing drone experiments. In subject experiments, the ethics committee for safety subjects it to an ethical review established by Cyber Media Center at Osaka University. From the Ethics Committee, the author obtained approval for data use in this subject. Appropriate care was taken to avoid any psychological or physiological distress on the subject during the experiment.

During the experiment, a user pointed to multiple windows on a building, and the UAV camera's image was recorded. A total of 6 individuals were able to participate in this setting. Participant age ranged from 23 to 31 years (M = 28.66, SD = 2.94), 1 participant was female, 50% of the participants had previous experience with manually flying a UAV, and 50% had previously used some form of gesture interface.

Each participant was positioned in 6 different spots from 12 possible spots (see Figure 5.18). These 12 spots were distributed in 3 columns and 4 rows in front of the building. The columns were aligned with the left, center, and right sides of the building. The closest row was 10 meters from the building, increasing a distance of 5 meters for every other row until the last row (25 meters from the building). For each spot, the user first opened both arms and pointed sideways, at no window, then the user pointed at ten windows, sequentially, 5 seconds for each window. A total of 432 combinations of windows, users and positions were tested.

To ensure the correct aim, the author attached a laser pointer (Figure 5.19) to the user's right arm. The laser was of 3000mW and 532 wavelengths so the user could point with precision, outdoors, in a long-range. In preliminary experiments, the author found substantial human error when trying to point without any visual feedback. So the author wanted to investigate integrating visual feedback from the UAV's POV so that the user can point more precisely and understand the output of the UAV's used system.

The UAV faced the building, with the user inside its view. Sometimes not all windows were present inside the UAV's image view. The UAV stood from 2 to 5 meters behind the user and 1.7 meters to 3 meters high. The author used this short distance because at longer distances, the 856 x 480 pixel image would

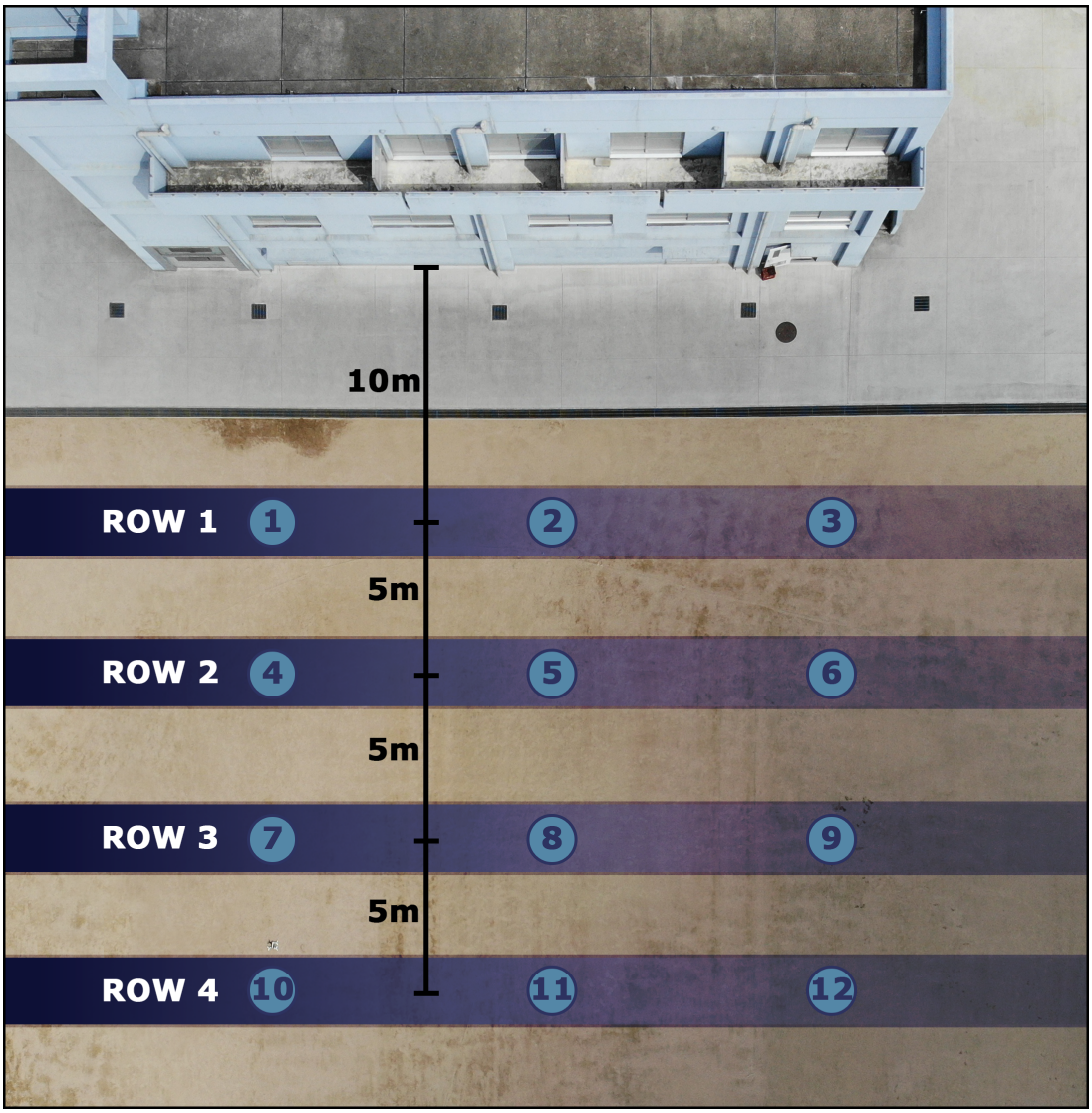


Figure 5.18. Top view of real-world experiment

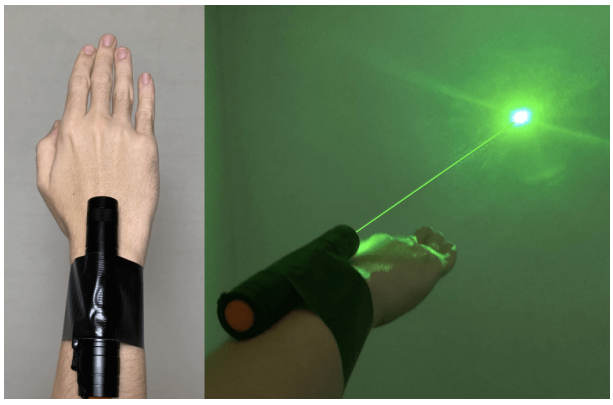


Figure 5.19. A Laser pointer was used in the experiments. As ground-truth, each user used a laser pointer to ensure they were pointing to the intended window.

lose details and cause OpenPOSE to output unsatisfactory results. By using a higher image resolution, this distance can be increased accordingly.

The author classified each pointed window as true positive if it was the correct window, as false positive if it wasn't the correct window, and as false negative if no window was selected. The author used true negative in two cases; the pointed window was outside the image, and the used approach determined that no window was selected. The second case of true negative was when the user was not pointing to any window.

A total of 432 combinations of windows, users, and positions were recorded and later processed using the two proposed approaches.

2D Pointing Gesture Interface Approach Result

This approach chooses the window with the center closest to the pointing line when crossing more than one window.

The results (Table 5.2) of the 432 combinations using the 2D approach showed an F1-Score of 0.45 and a Matthews Correlation Coefficient (MCC) of 0.1, which shows that this approach was only slightly better than the neutral points of both metrics. The experiment used an array of windows on the same plane, and this approach worked better when the targets were in different planes, due to the limitation of using only 2D data.

Table 5.2

Summary of Experiments on Real-World Environment

	2D R1	2D R2	2D R3	2D R4	2D All	3D R1	3D R2	3D R3	3D R4	3D All
Row-Bldg. Dist.	10m	15m	20m	25m	10-25m	10m	15m	20m	25m	10-25m
MCC	0.39	0.23	0.09	-0.26	0.1	0.53	0.19	0.1	-0.003	0.26
F1	0.65	0.48	0.39	0.28	0.45	0.81	0.78	0.70	0.58	**0.73**

Analyzing the results by row: The first row shows an F1-Score of 0.65 and an MCC of 0.39, the second row an F1-Score of 0.48 and an MCC of 0.23, the third row an F1-Score of 0.39 and an MCC of 0.09, and the fourth row an F1-Score of 0.28 and an MCC of -0.26. So the results worsen the more distant the row from the building, which is expected since the target area decreases with distance.



(a) (b)

Figure 5.20. (a) This figure shows the outputs of OpenPOSE, YOLO, and the 2D pointing gesture interface approach. This sample is a false positive; the correct window is above the selected one. This error is due to a line crossing multiple windows is ambiguous. This example is the most significant cause of error given the lack of 3D information to make a more informed decision in this approach. (b) This figure shows the outputs of OpenPOSE, YOLO, ORB-SLAM, and the 3D pointing gesture interface approach. Here, the calculation considers the estimated depth from projective transformations and monocular SLAM's point cloud, resulting in selecting the correct window. This sample is a true positive.

A sample result of this approach is in Figure 5.20(a), it is possible to see the outputs from OpenPOSE and YOLO. The result shown in the picture is a false positive. The user was pointing to the window above the selected one at that moment. This ambiguity caused by the lack of tree dimensional information in the decision process is the most significant cause of errors in the 2D approach.

3D Pointing Gesture Interface Approach Result

This approach uses ORB-SLAM to infer the real-worlds thirds dimension, from monocular images, to predict the pointed window better. The calibrated file used for each participant was previously recorded.

The results (Table 5.2) of the 432 combinations using the 3D pointing gesture interface approach showed an F1-Score of 0.73 and a Matthews Correlation Coefficient (MCC) of 0.26, which offers a significant improvement from the previous result of the 2D Approach.

Analyzing the results by row: The first row shows an F1-Score of 0.81 and



Figure 5.21. Samples from the 3D pointing gesture interface approach. (a) - (c) True positive samples. (d) - (f) False positive samples. (g) - (i) False negative samples. (j) - (l) True negative samples.

an MCC of 0.53, the second row an F1-Score of 0.78 and an MCC of 0.19, the third row an F1-Score of 0.70 and an MCC of 0.1, and the fourth row an F1-Score of 0.58 and an MCC of -0.003. So the results worsen the more distant the row from the building, which is expected since the target area decreases with distance.

A sample result of this approach is in Figure 5.20(b), it is possible to see the outputs from ORB-SLAM, OpenPOSE and YOLO. The result shown in the picture is a true positive.

The author classified each detected pointed window as true positive if it was the correct pointed window (Figure 5.21(a) - 5.21(c)). The author classified a detected pointed window as false positive if it wasn't the correct pointed window (Figure 5.21(d) - 5.21(f)). The author classified as false negative if no window was selected when the user was pointing to a window on the scene (Figure 5.21(g) - 5.21(i)). The author classified as true negative when either the correctly pointed window was outside the image, or the user was not pointing

to a window (Figure 5.21(j) - 5.21(l)).

5.5.3 Comparing Simulation and Real-World Experiment Conditions

The main differences between the simulation and real-world experiment is that the UAV can stay completely still in the simulation setting, and in the real-world experiment there will be some drift. The consequence is that in the simulation only one frame is needed to evaluate a given scenario. In the real-world, the author considers a four seconds sequence of frames to make a decision about a scenario.

Regarding the distance from the user to the building: The distance in the simulation rows were from a maximum of 17 meters from the building (row 1, Figure 5.15), to a minimum of 3 meters from the building (row 3, Figure 5.15). In the real-world experiment the maximum distance of a row was 25 meters from the building (row 4, Figure 5.18), and the minimum distance was 10 meters from the building (row 1, Figure 5.18).

Regarding the distance from the user to the UAV: The distance in the simulation was from 3 to 7 meters. In the real world scenario, that distance cannot be guaranteed due to drift, but it was around 5 meters. In the simulation, the UAV stood at a fixed distance of 2.5 meters from the ground. In the real-world, the UAV averaged 3.5 meters from the ground.

5.6 Discussion

The results from the 3D pointing gesture interface approach showed significant improvement in the limited setting in which it operated. Figure 5.22 shows the most significant improvement from the 2D approach to the 3D pointing gesture interface approach. With the 3D pointing gesture interface approach it is possible to distinguish between elements aligned on the same plane, or when there aren't any object being pointed inside the scene.

The UAV distance to the user was primarily determined by acceptable output from openPOSE, given the used resolution. The author understands that by increasing the resolution of the UAV image, it is possible also to increase the

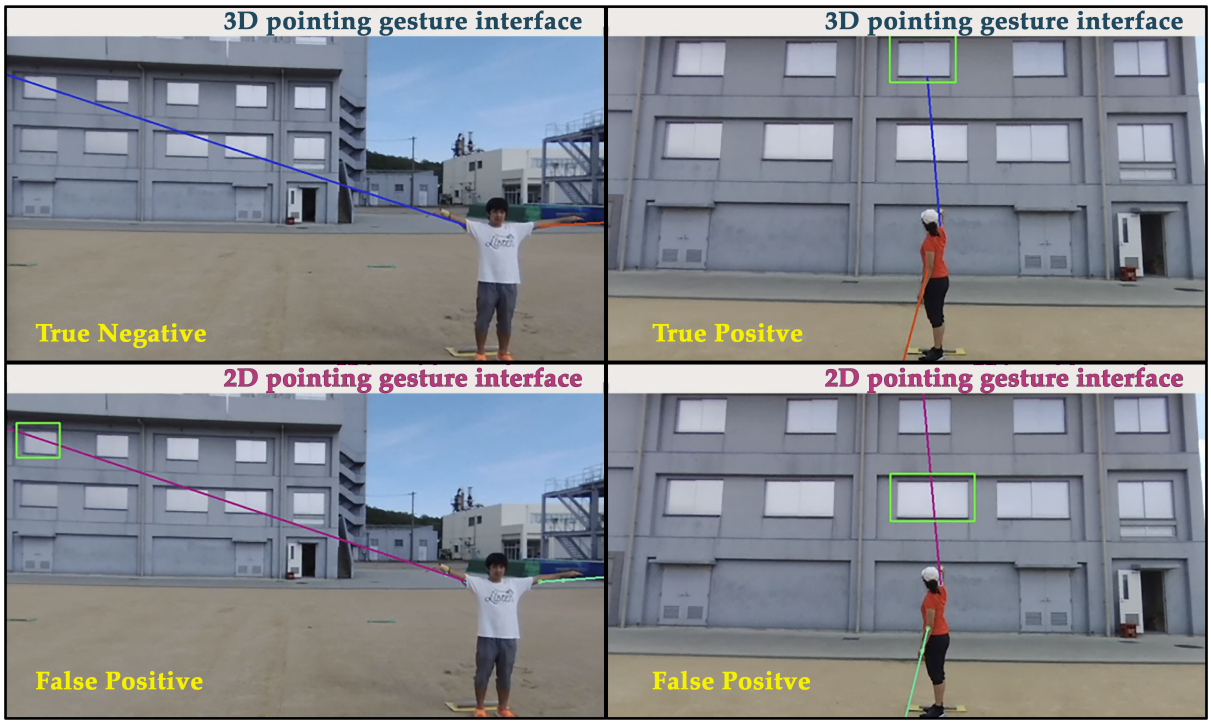


Figure 5.22. Samples of the 3D pointing gesture interface approach outperforming the 2D pointing gesture interface approach, given the same scene.

distance between the UAV and user, but it will also increase the computational cost of the system. When running the experiment, the SLAM based system ran at 16 fps on average, and the 2D based system ran at 29 fps.

The use of ORB-SLAM required at least a small amount of translational movement to keep updating the generating point cloud. Sometimes, the residual drift from the UAV could accomplish that; other times, the author had to translate the UAV by manual commands. The author noticed this because she was also running the ORB-SLAM when recording the UAV's camera topic. To obtain a point cloud the UAV was moved using sideways motion (translation), in a short space of around half meter around the UAV initial position on air.

The author also shows that the sparse point cloud from ORB-SLAM could detect the user position, and the translational movements ensured that the point cloud updated that position after a few moments. The output of ORB-SLAM was also incremented to provide the point cloud associated with the current image frame, and that is a modification to the original ORB-SLAM code, which

the author will make available for further research.

Occlusion occurred when the user's arm was not visible because other parts of the user body overlaid the pointing arm (Figure 5.23). Often this body part was the head. Occlusion occurred in less than 4% of the 432 combinations tested. It happened mostly in the middle column, that is, positions 2, 5, 8, and 11 of the experiment setting (Figure 5.18). This result suggests that pointing from the side columns resulted in better detection from the UAV's POV.

The author filtered the OpenPOSE output so that it would use the skeleton of only the user, even if other persons appeared in the background of the image. The author did that by filtering the biggest OpenPOSE skeleton fist, then tracking that skeleton using pixel distance from the previous frame. The result was that after the initial movement, it could keep track of the correct skeleton regardless of its size. For smoothness, the author used the averaged value of the last ten keypoints, for each OpenPOSE keypoint on the user's skeleton.

The calibration file for each person was previously recorded in another environment to simulate the real-world setting where one calibration file was used multiple times for the same person.

The laser pointer was used in the real-world experiment to ensure that the participants are pointing accurately to the correct window. The 3D pointing gesture interface approach does not rely on the laser pointer. In a multitude of scenarios, it is unrealistic to expect the user to always have a laser pointer in handy. In future works, if the UAV shows where the human is pointing, it is possible to use that visual representation of the area of interest to select objects and guide the UAV. So if the UAV has some kind of projection, or laser, it projects on the humans tracked pointing area and, even if there are accuracy errors, seeing that representation, the human can correct the pointing tracked area to the desired place.

The current 3D pointing gesture interface approach could be used for other scenarios besides firefighting, it has a variety of applications because it's based only on a monocular camera, so the only constraint is the need for a monocular camera, a computational unit and a wireless signal. The "select" gesture is very general, it's for example a good cue for intent, and there are many lines of re-



Figure 5.23. Sample of OpenPOSE's output where occlusion occurred. On both images, the head overlaid the pointing arm. On the top image, the hand's keypoint could not be "guessed." On the bottom image, even with occlusion, the hand keypoint was "guessed" close to the correct position.

searches that try to understand human intent [72, 73, 74]. For example, this can be used to understand human intent though camera security on airports, where seeing where people are pointing helps raise security concerns, depending on other factors. Other application example could be a companion UAV that can interact with ordinary objects selected by the user. It could also be used in an educational environment, in outdoor classes when the professor points towards the surroundings, the class camera can identify the target and show a specific content for the class.

One limitation of the system presented here is that in the real-world experiment, the UAV was facing the building as statically as possible, although a considerable amount of drift was present due to wind conditions at the time. Another limitation is that the user must always be inside the field of view of the UAV, to give commands to the UAV. So there must be a confirmation before the drone moves to the desired target, possibly losing sight of the user.

5.7 Conclusion

The Gesture Development Process presented in Chapter 3 resulted in a gesture for "select" (Figure 3.13). Because of the wide-spread applicability of the "select" system function, the author chose this system function and associated gesture

Given its wide applicability, the author chose the system function "select" and its associated gesture (Figure 3.13) to explore multi-scale gestural interactions.

Given the wide variety of applications and multiple degrees of freedom, UAVs were chosen as a means to explore the "select" system function in a multi-scale gestural interaction scenario. In past works, the author analyzed which gesture fits best for a particular scenario, used our Gesture Development Process [55], validated it in general tasks and with a diverse group of users [75], and conducted an elicitation study with the targeted users [76].

In this chapter the author presented a system that enables UAVs equipped with a monocular camera to determine which window of a building was selected

by a human user, on a large-scale, indoors or outdoors. The author developed two applications: one with monocular SLAM and another without SLAM (no depth information). Experimental evaluation showed that the 3D pointing gesture interface obtained, on average, a 0.85 F1-Score in simulation, and a 0.73 F1-Score in a real-world setting. The 3D pointing gesture interface obtained a 0.58 F1-Score when considering only the results obtained at the maximum distance of 25 meters between drone and building.

5.7.1 Contributions

The contributions of the work presented in this chapter were:

- Elicitation study on firefighters' experience in a first-response scenario.
- Accessible multi-scale monocular-based 2D and 3D pointing gesture interface proposed applications.
- Modification of ORB-SLAM's output to include the point cloud of only the elements appearing in the current frame.
- Verification and validation on simulated and real-world (large-scale) fire brigade scenario.
- A dataset of pointing to windows outdoor.

Chapter 6

Conclusion

As described in Section 1.7, this final chapter presents a summary of the main findings of this research project and provides some suggestions regarding future studies.

6.1 Summary of Major Findings

The aim of this research was to investigate the design, evaluation and deployment of gestural interfaces. The author reviewed many gesture design processes and then designed a detailed Gesture Development Process that takes into account the most effective approaches found in literature, and addressed shortcoming in past works. The result is Gesture Development Process with a innovative cyclic flow and thoroughly documented artifacts produced during the application of the process. This documentation can be specially useful in collaborative works and reuse of previous results.

The presented Gesture Development process serves to guide designers through the development of user interfaces based on gestures. The author demonstrated the Gesture Development Process by creating gestures for five system functions: “Select”, “Rotate”, “Scale”, “Translate”, and “Stop”. The Final Gesture Vocabulary was then deployed in an application named CubeNI (for Natural Interaction), using the Leap Motion device as gestural interface input. To further validate and evaluate the proposed Gesture Development Process the author then conducted tests on the CubeNI in multicultural conditions, with participants from Japan and Brazil.

Another aim of the multicultural evaluation proposed by the author was to analyze how different technological familiarization levels impacts the perceived user experience of gestural interfaces. First, the author set up an experiment to compare a gestural interface (CubeNI) with a mouse interface (CubeMK). This experiment took place inside university environments of Japan and Brazil. The AttrakDiff questionnaire was used to evaluate user experiences of both products.

The results showed that statistically significant differences were found only in two situations: between Brazil and Japan regarding CubeMK's results, and between CubeNI and CubeMK's results in Japan (Figure 4.11). As for the four qualities group (Figure 4.12), in Brazil, the comparison between CubeNI's and CubeMK's HQ-S and ATT qualities had statistically significant results. In Japan, the same goes for the comparison between CubeNI's and CubeMK's HQ-S and ATT qualities. Additionally, the comparison between Brazil-CubeMK's and Japan-CubeMK's ATT qualities also had statistically significant results.

In both countries the major advantage of the gestural interface was that it was considered much more stimulating to the users' senses than the mouse-based interface. The disadvantage was that the pragmatic qualities of the gesture interface has yet to match the mouse's objectiveness. This suggests that there is still a need to improve the ease of use of gesture interfaces, so that the users' attention can focus entirely on task completion and not on interface usage. To that end it is recommended improving feedback when system is unable to recognize user's gestures. Also, CubeNI's Gesture Vocabulary was also validated in Japan.

Regarding gestural interfaces many paths could be taken to contribute to its development: Researching how to improve metrics for evaluating this type of interface would be welcomed, more studies in different countries can be beneficial to better understand what is globally expected from this type of interface, and research on gesture precision enhancement for future gestural interfaces could help to increase this type of interface's pragmatic qualities. To that end, during the course of tests, improvement on feedback was suggested when the system failed to recognize the gestures. Other possible continuation to this investigation course is measuring how users perform (time, effort, accuracy) when

given more complex test routines, in addition to perception investigation, in order to balance subjectiveness with objectiveness.

Chapter 4 of the present thesis was aimed at short distances interaction, that is, short-scale interactions. Chapter 5 proposed to include the investigation of gestural interaction up to much larger environments, that is, multi-scale interactions.

When the user is interacting with objects 10, 15, 25 meters away, the slightest change in the computer vision perception of the users movements can lead to erroneous results. The author also believes that accessible computer vision solutions should be made available as much as possible, promoting inclusion in all areas of research. It was in this challenging scenario that the author proposed accessible multi-scale monocular-based 2D and 3D pointing gesture interfaces.

The main objective of the 3D pointing gesture interface is to present a system that enables UAVs equipped with a monocular camera to determine which window of a building was selected by a human user, on a large-scale, indoors or outdoors. The author developed two applications: one with monocular SLAM and another without SLAM (no depth information). Experimental evaluation showed that the 3D pointing gesture interface obtained, on average, a 0.85 F1-Score in simulation, and a 0.73 F1-Score in a real-world setting. The 3D pointing gesture interface obtained a 0.58 F1-Score when considering only the results obtained at the maximum distance of 25 meters between drone and building.

The author's contributions include a Human-Drone Interaction investigation on firefighting's first response needs, and the large-scale monocular-based 2D and 3D pointing gesture interface approaches. The author also contributed to the modification of ORB-SLAM's output to include the current frame's respective point cloud, made available for further research. Finally, the verification and validation of both approaches on the simulated environment and real-world scenario contributed by producing artifacts such as a dataset of humans pointing to windows outdoor.

In the future, the author hopes that the relational data generated in the artifacts of the Gesture Development Process creates a dataset for future non-human dependant Gesture Development Processes.

Bibliography

[1] Gaines, B.R., "From ergonomics to the Fifth Generation: 30 years of human-computer interaction studies," *Computer Compacts*, vol. 2, no. 5-6, pp. 158–161, 1984.

[2] Gould, J.D. and Lewis, C., "Designing for usability: key principles and what designers think," *Communications of the ACM*, vol. 28, no. 3, pp. 300–311, 1985.

[3] M. Chu and B. Begole, "Natural and implicit information-seeking cues in responsive technology.," in *Human-centric interfaces for ambient intelligence.* (A. J. Aghajan, H. and R. e. Delgado, eds.), ch. 17, pp. 415–452, Academic Press., 2010.

[4] Desai, P.R., Desai, P.N., Ajmera, K.D. and Mehta, K., "A review paper on oculus rift-a virtual reality headset," *arXiv preprint arXiv:1408.1173*.

[5] Kelly, J.W., Cherep, L.A. and Siegel, Z.D., "Perceived space in the HTC Vive," *ACM Transactions on Applied Perception (TAP)*, vol. 15, no. 1, pp. 1–16, 2017.

[6] Katona, J., Farkas, I., Ujbanyi, T., Dukan, P. and Kovari, A., "Evaluation of the NeuroSky MindFlex EEG headset brain waves data," *IEEE 12th international symposium on applied machine intelligence and informatics (SAMI)*, pp. 91–94, 2014.

[7] Ye, M., Zhang, Q., Wang, L., Zhu, J., Yang, R. and Gall, J., "A survey on human motion analysis from depth data," *Time-of-flight and depth imaging. sensors, algorithms, and applications*, pp. 149–187, 2013.

[8] Wexelblat, A., "An approach to natural gesture in virtual environments.," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 2, no. 3, pp. 179–200, 1995.

[9] Lopatovska, I., Rink, K., Knight, I., Raines, K., Cosenza, K., Williams, H., Sorsche, P., Hirsch, D., Li, Q. and Martinez, A., "Talk to me: Exploring user interactions with the Amazon Alexa," *Journal of Librarianship and Information Science*, vol. 51, no. 4, pp. 984–997, 2019.

[10] López, G., Quesada, L. and Guerrero, L.A., "Alexa vs. Siri vs. Cortana vs. Google Assistant: a comparison of speech-based natural user interfaces," *International Conference on Applied Human Factors and Ergonomics*, pp. 241–250, 2017.

[11] Galeso, M., "Apple Siri for Mac: An Easy Guide to the Best Features,"

[12] Almeida, F.B.D., "Sistema interativo baseado em gestos para utilização de comandos no computador," 2013.

[13] Bacim, F., Nabiyouni, M. and Bowman, D.A., "Slice-n-Swipe: A free-hand gesture user interface for 3D point cloud annotation," *IEEE Symposium on 3D User Interfaces (3DUI)*, pp. 185–186, March 2014.

[14] Jeong, S., Jin, J., Song, T., Kwon, K. and Jeon, J.W., "Single-camera dedicated television control system using gesture drawing," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 4, pp. 1129–1137, 2012.

[15] Nielsen, M., Störring, M., Moeslund, T.B. and Granum, E., "A procedure for developing intuitive and ergonomic gesture interfaces for HCI," *International gesture workshop, Springer*, pp. 409–420, April 2003. Berlin, Heidelberg.

[16] Nielsen, M., Moeslund, T. B., Storring, M. and Granum, E., "Gesture interfaces," in *HCI Beyond the GUI: Design for Haptic, Speech, Olfactory and Other Nontraditional Interfaces*, ch. 3, pp. 75–103, Morgan Kaufmann, 2008.

[17] Henze, N., Löcken, A., Boll, S., Hesselmann, T. and Pielot, M., "Free-hand gestures for music playback: deriving gestures with a user-centred process," *Proceedings of the 9th international conference on Mobile and Ubiquitous Multimedia*, pp. 1–10, December 2010.

[18] Epps, J., Lichman, S. and Wu, M., "A study of hand shape use in table-top gesture interaction," *CHI'06 extended abstracts on human factors in computing systems*, pp. 748–753, April 2006.

[19] Choi, E., Kwon, S., Lee, D., Lee, H. and Chung, M.K., "Towards successful user interaction with systems: Focusing on user-derived gestures for smart home systems," *Applied ergonomics*, vol. 45, no. 4, pp. 1196–1207, 2014.

[20] Jeong, S., Jin, J., Song, T., Kwon, K. and Jeon, J.W., "Single-camera dedicated television control system using gesture drawing," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 4, pp. 1129–1137, 2012.

[21] Wu, H., Wang, Y., Liu, J., Qiu, J. and Zhang, X.L., "User-defined gesture interaction for in-vehicle information systems," *Multimedia Tools and Applications*, vol. 79, no. 1, pp. 263–288, 2020.

[22] Nakakoji, K., "Beyond language Translation: Crossing the Cultural Divide," *IEEE Software*, vol. 13, no. 6, pp. 42–46, 1996.

[23] de Wet, L., Blignaut, P. and Burger, A., "Comprehension and usability variances among multicultural web users in South Africa," *CHI Extended Abstracts on Human Factors in Computing Systems*, pp. 526–527, 2002.

[24] Mitra, S., and Acharya, T., "Gesture Recognition: A Survey.," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 3, p. 311–324, 2007. https://doi.org/10.1109/TSMCC.2007.893280.

[25] Van den Bergh, M., and Van Gool, L. , "Combining RGB and ToF Cameras for Real-Time 3D Hand Gesture Interaction," *IEEE Workshop*

on Applications of Computer Vision (WACV), p. 66–72, January 2011. https://doi.org/10.1109/ WACV.2011.5711485.

[26] Nandakumar, R., Kellogg, B., and Gollakota, S., "Kinect sensor-based long-distance hand gesture recognition and fingertip detection with depth information," *Journal of Sensors*, 2018.

[27] Choi, J. W., Ryu, S. J., and Kim, J. H., "Short-range radar based real-time hand gesture recognition using LSTM encoder," *IEEE Access*, vol. 7, pp. 33610–33618, 2019.

[28] Dankovich, L. J., and Bergbreiter, S., "Gesture Recognition via Flexible Capacitive Touch Electrodes," *International Conference on Robotics and Automation (ICRA)*, pp. 9028–9034, May 2019.

[29] Chossat, J. B., Tao, Y., Duchaine, V., and Park, Y. L., "Wearable soft artificial skin for hand motion detection with embedded microfluidic strain sensing," *IEEE international conference on robotics and automation (ICRA)*, pp. 2568–2573, May 2015. https://doi.org/10.1109/ICRA.2015.7139544.

[30] DelPreto, J., and Rus, D., "Sharing the load: human-robot team lifting using muscle activity," *International Conference on Robotics and Automation (ICRA)*, pp. 7906–7912, May 2019.

[31] DelPreto, J., and Rus, D, "Plug-and-Play Gesture Control Using Muscle and Motion Sensors," *ACM/IEEE International Conference on Human-Robot Interaction*, pp. 439–448, March 2020.

[32] Kim, J., Mastnik, S., and André, E. , "EMG-based hand gesture recognition for realtime biosignal interfacing," *International Conference on Intelligent User Interfaces*, pp. 30–39, January 2008.

[33] Samadani, A. A., and Kulic, D., "Hand Gesture Recognition Based on Surface Electromyography," *International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 4196–4199, August 2014.

[34] Nandakumar, R., Kellogg, B., and Gollakota, S., "Wi-fi gesture recognition on existing devices," *arXiv preprint arXiv:1411.5394.*, 2014.

[35] Tölgyessy, M., Dekan, M., Duchoň, F., Rodina, J., Hubinský, P., and Chovanec, L. U., "Foundations of visual linear human–robot interaction via pointing gesture navigation," *International Journal of Social Robotics*, vol. 9, no. 4, pp. 509–523, 2017.

[36] Liu, T., Chen, Z., and Wang, X., "Automatic Instructional Pointing Gesture Recognition by Machine Learning in the Intelligent Learning Environment," *International Conference on Distance Education and Learning*, pp. 153–157, May 2019.

[37] Gromov, B., Gambardella, L.M. and Giusti, A., "Video: Landing a drone with pointing gestures," *Companion of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 374–374, 2018.

[38] Azari, B., Lim, A. and Vaughan, R., "Commodifying Pointing in HRI: Simple and Fast Pointing Gesture Detection from RGB-D Images," *16th Conference on Computer and Robot Vision (CRV)*, pp. pp. 174–180, 2019.

[39] Mirri, S., Prandi, C. and Salomoni, P., "Human-Drone Interaction: state of the art, open issues and challenges," *Proceedings of the ACM SIGCOMM 2019 Workshop on Mobile AirGround Edge Computing, Systems, Networks, and Applications*, pp. 43–48, 2019.

[40] Obaid, M., Kistler, F., Kasparavičiūtė, G., Yantaç, A.E. and Fjeld, M., "How would you gesture navigate a drone? a user-centered approach to control a drone," *Proceedings of the 20th International Academic Mindtrek Conference.*, pp. 113–121, 2016.

[41] "Lidar sensors for robotic applications." https://www.sentekeurope.com/robotics-lidar.

[42] Foley, J.D., Van, F.D., Van Dam, A., Feiner, S.K., Hughes, J.F., Angel, E. and Hughes, J., *Computer graphics: principles and practice*. Addison-Wesley Professional, 1996.

[43] Arhippainen, L., "Capturing user experience for product design," *The 26th*, pp. 1–10, 2003.

[44] D'Hertefelt, S., "Emerging and future usability challenges: designing user experiences and user communities," *InteractionArchitect.com*, p. 2, 2000.

[45] User Interface Design GmbH (UID), "AttrakDiff." http://www.attrakdiff.de/sience-en.html.

[46] Hassenzahl, M., "Beautiful objects as an extension of the self: A reply," *Human-Computer Interaction*, vol. 19, no. 4, pp. 377–386, aa.

[47] Hassenzahl, M., "Hedonic, emotional, and experiential perspectives on product quality," *Encyclopedia of human computer interaction*, pp. 266–272, aa.

[48] Hassenzahl, M., Tractinsky, N., "User experience-a research agenda," *Behaviour & information technology*, vol. 25, no. 2, pp. 91–97, 2006.

[49] Tractinsky, N., Hassenzahl, M., "Arguing for aesthetics in human-computer interaction," *I-com*, vol. 4, no. 3, pp. 66–68, aa.

[50] Schalock, R.L., *Outcome-based evaluation.* Springer Science Business Media, 2001.

[51] Hassenzahl, M., "The thing and I: understanding the relationship between user and product," *Funology, Springer*, pp. 31–42, 2003.

[52] Hassenzahl, M., "Experience design: Technology for all the right reasons," *Synthesis Lectures on Human-Centered Informatics*, pp. 1–95, 2010.

[53] ITU(2017), "Measuring the Information Society Report 2017: Volume 1 and 2." http://handle.itu.int/11.1002/pub/80f52533-en.

[54] ITU(2016), "Measuring the Information Society 2016." http://handle.itu.int/11.1002/pub/80de56e8-en.

[55] Medeiros, A. C., Tavares, T. A., and da Fonseca, I. E., "How to Design an User Interface Based on Gestures?," *International Conference of Design, User Experience, and Usability (Springer, Cham.)*, pp. 63–74, 2018.

[56] Reich, L., "How Drones are being used in Disaster Management?." http://geoawesomeness.com/drones-fly-rescue/, 2016.

[57] Azevedo, M. A., "Drones give journalists a new way to report news." https://newsroom.cisco.com/feature-content?articleId=1851973, 2017.

[58] Smith, S., "Military and Civilian Drone Use - The Future of Unmanned Aerial Vehicles.." www.thebalancecareers.com/military-and-civilian-drone-use-4121099, 2017.

[59] Funk,M. , "Human-drone interaction: let's get ready for flying user interfaces!," *ACM Interactions*, vol. 25, pp. 78–81, 2018.

[60] Kim, J., Lee, Y.S., Han, S.S., Kim, S.H., Lee, G.H., Ji, H.J., Choi, H.J. and Choi, K.N., "Autonomous flight system using marker recognition on drone," *21st Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV) IEEE*, pp. 1–4, 2015.

[61] Al-Eidan, R.M., Al-Khalifa, H. and Al-Salman, A.M., "A Review of Wrist-Worn Wearable: Sensors, Models, and Challenges.," *Journal of Sensors*, 2018.

[62] Alsheakhal, M., Skaik, A., Aldahdouh, M., Alhelou, M., "Hand Gesture Recognition System," *Information & Communication Systems*, p. 132, 2011.

[63] Redmon, J. and Farhadi, A., "YOLO9000: better, faster, stronger," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

[64] Cao, Z., Hidalgo, G., Simon, T., Wei, S.E. and Sheikh, Y., "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[65] Redmon J, Farhadi A., "Yolov3: An incremental improvement," *arXiv preprint*, 2018. arXiv:1804.02767.

[66] Goldman, R., "Intersection of two lines in three-space," *Graphics Gems*, 1990.

[67] Mur-Artal, R. and Tardós, J.D., "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[68] Mur-Artal, R. and Tardós, J.D., "ORB-SLAM: tracking and mapping recognizable features," *Workshop on Multi View Geometry in Robotics (MVIGRO)*, vol. 2014, p. 2, 2014.

[69] Koenig, N. and Howard, A., "Design and use paradigms for gazebo, an open-source multi-robot simulator," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, pp. 2149–2154, September 2004.

[70] "Parrot Bebop 2 drone." www.parrot.com/us/drones/parrot-bebop-2.

[71] "Robot Operating System (ROS) - Documentation." http://wiki.ros.org/Documentation, 2020.

[72] Jin, Z. and Pagilla, P.R., "Human-robot teaming with human intent prediction and shared control," *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, vol. 11413, no. 2, p. 114130X, 2020.

[73] Holtzen, S., Zhao, Y., Gao, T., Tenenbaum, J.B. and Zhu, S.C., "Inferring human intent from video by sampling hierarchical plans," *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pp. 1489–1496, 2016.

[74] Erden, M.S. and Tomiyama, T., "Human-intent detection and physically interactive control of a robot without force sensors," *IEEE Transactions on Robotics*, vol. 26, no. 2, pp. 370–382, 2010.

[75] Medeiros, A. C. S., Ratsamee, P., Uranishi, Y., Mashita, T., Takemura, H., and Tavares, T. A., "3D Gesture Interface: Japan-Brazil Perceptions,"

International Conference on Human-Computer Interaction (Springer), pp. 266–279, July 2020.

[76] Medeiros, A. C., Ratsamee, P., Uranishi, Y., Mashita, T., and Takemura, H., "Human-Drone Interaction: Using Pointing Gesture to Define a Target Object," *International Conference on Human-Computer Interaction (Springer)*, pp. 688–705, July 2020.