FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Tracking and Ground Truth system for mobile robots

Francisco de Castro e Costa

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Hélio Sousa Mendonça

Second Supervisor: Carlos Miguel Correia da Costa

July 26, 2019

Resumo

Métodos de localização de robôs são de crescente importância em várias áreas, tanto num ambiente experiental como industrial. No entanto, soluções que determinem com precisão a localização e orientação de um robô são normalmente dispendiosas e requerem um grande número de sensores. O sistema *SteamVR* foi estudado com o objetivo de determinar se este é capaz de fornecer uma precisão desejada, de modo a ser usado como método alternativo de localização. Para além disso, foi avaliada a possibilidade deste sistema ser utilizado em duas aplicações principais: robots móveis, nomeadamente de futebol robótico, e robots industriais.

Para isso, foi desenvolvido um suporte capaz de permitir deteção pelo sistema *SteamVR*, e foram feitos testes, comparando este sistema com um de precisão conhecida. Sob vários pârametros diferentes, os dois sistemas foram comparados, sendo os resultados registados e comparados. A conclusão principal tomada confirma a possibilidade de o método ser utilizado para localização de robôs móveis, assim como maior parte de robôs industriais com necessidade de precisão de ordem milimétrica.

Abstract

Robot tracking methods are of rising importance in various areas, either in an experimental environment, or in an industrial one. However, solutions that are capable of tracking the location and orientation of a robot are usually expensive and require a large amount of external sensors. In order to explore alternative methods of tracking, SteamVR was studied with the objective of determining this system's precision. Moreover, the possibility of this system being used on two applications was evaluated: mobile robots, specifically related to robotic football, and industrial robots.

For that, a support that was able to be detected by the SteamVR system was developed. Afterwards, tests were conducted, comparing it to another system of known precision. Under various different parameters, both systems were compared, with the results documented and compared. The main conclusion taken confirms the possibility of the method being used for tracking of mobile robots, as well as most mobile robots, with millimetrical precision requirements.

Acknowledgements

I'd like to thank my supervisor, Hélio Sousa Mendonça, for his guidance on this thesis. His constant orientation during the whole duration of the thesis was very helpful, and his persistence with me allowed this document to reach the state it has. I'm thankful for his dedication and I hope I was able to match his expectations in the end.

Thanks to my co-supervisor, Carlos Miguel Correia da Costa, for his help during this semester. He helped solve some tough problems with his advice that would have been difficult to figure out on my own, and his time and expertise allowed for this thesis to progress more smoothly than it would have otherwise.

Thanks to Eurico Sousa, Joana Dias, Pedro Melo, André Castro, Pedro Relvas, Cláudia Rocha, between other members of INESC-TEC, for their availability and eagerness to help.

Finally, I'd like to thank my family for the constant help and support throughout all the years of studying.

Francisco de Castro e Costa

Contents

List of Figures

List of Tables

Abbreviations and Symbols

CCD	Charge Coupled Device
CMOS	Complementary Metal-Oxide Semiconductor
DoF	Degrees of Freedom
EKF	Extended Kalman Filter
HDK	Hardware Development Kit
HMD	Head Mounted Device
HWID	HardWare IDentification
IMU	Inertial Measurement Unit
IR	InfraRed
JSON	JavaScript Object Notation
LED	Light Emitting Diode
LIDAR	LIght Detection And Ranging
mm	millimeter(s)
PC	Personal Computer
RGB	Red, Green and Blue
STD	STandard Deviation
STL	STereoLithography
VGA	Video Graphics Array

Chapter 1

Introduction

This chapter introduces the main context of the thesis, explaining the motivation behind it and what is planned to be achieved, as well as outlining the contents of the rest of the document.

1.1 Context

Robot tracking is an extremely important matter in mobile and industrial robotics; a robot needs to know its location to efficiently execute its tasks. Some of the systems used currently can be used as Ground Truth in robot tracking, given their higher precision overs traditional, LIDAR-based methods. However, this are expensive, as they use several external sensors to obtain a significant workspace area, complicating widespread use. Using the recent SteamVR technology, it's possible to develop a tracking system for a significantly lower price.

1.2 Motivation

This problem was motivated by the possibility of implementing a system capable of estimating the 6DoF pose of a mobile robot for use in the context of robotic football, as a way of establishing an alternative to other tracking methods, with more accessible pricing without sacrificing accuracy or sensor complexity and thus satisfying the need for a precise, flexible tracking system, which can work simultaneously with several robots.

1.3 Goals

The objectives of this thesis can be divided in two points: implementation of a prototype of motion tracking based on SteamVR, and determination and optimization of the geometric configuration of the sensors. Through this, its the main goal to reach conclusions about applications of the SteamVR system in mobile and industrial robotics.

1.4 Thesis Structure

Other than the introduction, this thesis contains 5 other chapters. In chapter 2, the state of the art and other relevant projects are analysed. Chapter 3 delves into SteamVR, explaining what it is and how it works. Chapter 4 follows the procedure of creating a tracked object to be detected by SteamVR from scratch. In chapter 5 are all the tests and observations on the precision of SteamVR. Chapter 6 delves into all the challenges and problems that arose in the making of this thesis. Finally, in chapter 7 conclusions are made about the precision of SteamVR, and its applicability on the real world.

Chapter 2

State of the Art

This chapter contains the research of the state of the art, identifying which are the solutions frequently used or available on the market, so they can be used as comparison.

2.1 Introduction

The specific search terms were divided in two groups: mobile robot tracking (robots capable of movement with the goal of interaction inside a certain environment) and industrial robot tracking (robots used mainly for manufacturing). Considering the theme of mobile robots, the search was more focused around the topic of robotic football, since one of the desired results was use of the system for this purpose. Some search into 6 DoF tracking was also made.

The main points of comparison with the presented solution were the more popular tracking systems, Optitrack and VICON. These compose mainly of high speed cameras along with tracking software, allowing for high precision in the tracking of robots. However, the main limitations of these systems are not only their high price, but the use of a higher number of sensors when compared to SteamVR.

2.2 Mobile Robots

One of the methods frequently used for robot tracking is based on triangulation [5], that is, in the use of three distance cameras which when combined allow for the detection of an object from the intersection of the circumferences centered around them, and with a radius equal to the detected distance. This method needs precise sensors for an acceptable precision, in conjunction of the disadvantage of needing 3 cameras total.

2.2.1 Optitrack

As the main point of comparison with SteamVR, *Optitrack* is a high precision tracking system, capable of covering a 6x6m workspace (using the Flex 13 cameras). It provides errors lower than 1mm, and as such is used as Ground Truth to measure the precision of SteamVR

The Optitrack system is usually composed of 6 IR cameras distributed around the area where tracking is desired. These cameras, usually placed at different heights in order to capture varying views of the object, are used in conjunction with small reflective spheres, which are placed around the object. These spheres reflect the IR light emitted by the cameras, and with that a camera is able to know the distance of that specific marker to it.

With two cameras' readings, the possible positions of the marker can be cut down to two points; however, since one of them always lays outside of the workspace, if two cameras are able to see a marker, then that marker's position is known. Of course, if additional cameras can see that marker, the overall tracking error is reduced. By tracking three points, you can also get the rotation of an object, getting the full 6DoF pose. However, an object has to have at least four markers in order to be tracked, and those markers must not be on the same plane.

2.2.2 Robotic Football

On the topic of robotic football, a good amount of tracking systems which have been used for a long time includes a camera positioned above the field, with the detection of robots being based on colors attributed to the robots [6].

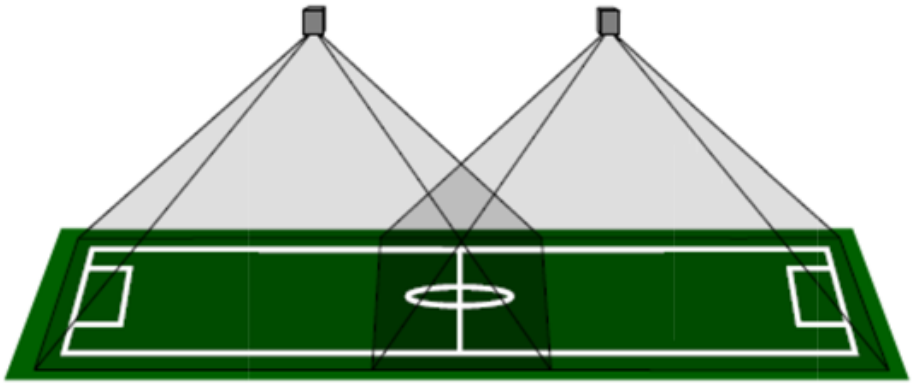

Figure 2.1: Example of a camera-based futebol robot tracking system

Given the rules of the RoboCup, the color for the field and walls are stipulated as green and white respectfully. Additionally, robots must have a blue or yellow circular area on top, depending on what team their in. Using this information, the robots position can be determined by detecting their respective color with the camera.

Given all the robots' positions, there's a need to determine their orientation. This is solved by placing a second, pink patch on top of the robot, a set distance away from team indicating one. The distance to the team patch identifies the robot it belongs to, and then its position is used to determine their orientation.

Lastly, there's the problem of matching the measures with each robot. One of the solutions to this case is to give a distinct color to each robot; however, this task is made difficult by the fact that a lot of colors are already in use (green, white, pink, ...), and thus the available amount of colors

is small. Other solution is to not color code the robots, and instead use a greedy algorithm based on a minimum distance criterion. While reliable, this method is very susceptible to noise, which might have the algorithm make erroneous matches.

Both of these solutions, while efficient and robust, not only need high processing power, but have a limited number of robots that can be tracked at any time, especially compared with the method being studied, where a virtually unlimited number of objects are supported.

Other solution found consisted also on using a camera as a sensor, but this time the tracked element wasn't colors but blinking Light Emitting Diodes (LED) at a known frequency [7].

Instead of a global sensor above the field, there are also tracking systems that give each robot a laser, which are used in conjunction with an Extended Kalman Filter to estimate its position based on goals spread around the field. These goals can also be distinguished with different colors, to facilitate their detection.

The EKF (Extended Kalman Filter) is an algorithm which, taking a set of measurements obtained during a certain amount of time, can provide estimations of variables even if influenced by noise or imperfections. In this case, it takes the measurements made, namely the readings from the wheels' odometry and the information from the laser, and outputs a prediction on the robots position taking into acount all previous readings. This laser information, in order to make sure it pertains to the goals, must be properly filtered, for example by limiting the distance where a reading is considered valid, or by only accepting results above a certain reflexion index, depending on the goal's material.

The main differences when compared with the studied solution are the bigger degree of errors, as well as the possibility of one or more goals being obstructed by other robots, affecting tracking.

Other methods exist other than these; however, there was an effort for finding solutions that allowed errors around 1mm, and the remaining solutions surpassed this parameter.

2.2.3 Kinect

Like SteamVR, Microsoft Kinect was also primarily designed for use in videogames; however, it is also widely used for robot tracking [8]. While it does contain a color VGA video camera, it is only used for aiding in face recognition and other features. Instead, it uses an infrared projector and a monochrome CMOS sensor in order to detect and identify the surrounding environment. This allows Kinect to track objects given any lighting condition, and with errors comparable to SteamVR's (around 1mm).

When moved to a new location, this system requires calibration of the play area, not unlike SteamVR, however, while the latter can work uncalibrated when using a single base station, Kinect always has to calibrate when in a new workspace. Additionally, SteamVR's physically placed sensors allow for more robust tracking and smaller processing time when compared to Kinect's detection system. Lastly, while the difference in price isn't dramatic, SteamVR still is the cheapest of the two by around 50 euros.

The Kinect is no longer supported by Microsoft,

Figure 2.2: Inside a Kinect controller [1]

2.3 Industrial Robots

Tracking for industrial robots has precision becoming a much more important requirement: because of the need of performing tasks with little margin for errors, the precision needed is usually around 0,20mm

The most basic solution of the ones found relied on high precision encoders. However, as expected, there's an increase in price with quality, and also the individual installation of each encoder sometimes makes it impossible to use this method for every case.

One of the methods found is very similar to SteamVR: a stationary laser mounted on a servo, and reflectors placed on the tracked robot, allow for robot tracking based on time-of-flight in conjunction with angular encoders [9]. While they're very similar, this method doesn't used sensors placed on the robot, opting instead for reflective tape, plus the added costs with laser quality and time-of-flight sensors raise the price considerably.

Other found solution allows the use of two sensors: a CCD camera (charge-coupled device), used to detect an object and keep it in view, along with a laser sensor, which function is to detect other objects and prevent collisions [10]. This solutions, compared to the on in study, provides similar results: while it has better precision, the difference in price is obvious, especially in this case since 2 sensors are needed instead of one.

The solution found at [11] is based on LEDs. In this case, twenty high intensity, differently colored LEDs are placed on each face of an icosahedron, and are tracked by two cameras. The high number of sensors and cameras are used to make sure every possible pose is covered, with no possibility of occlusions. Problems with noise are solved by flashing for a short amount of time: the high intensity of the light drowns out surrounding noise and helps the reading be precise.

When talking about industrial manipulators, the issue becomes not how to know the location of the robot, but how accurately the robot is being controlled. In this application, the task becomes

to enhance the accuracy of the industrial robots, many times by actively calibrating their position. In [12] a laser tracker is used, in conjunction with software that takes into account all possible errors, as well as joint stiffness. Laser trackers are also used in [13] ,but this time accompanied with a Kalman filter instead. Thought popular, these kinds of systems are often either expensive, if focused on precision, or slow, if focused on price, with compromises having to be made between the two.

Found in [2] is an alternative to laser trackers. This one is based on optical CMMs, or Co-ordinate Measuring Machines. Optical CMMs calculate a model's pose through image based triangulation, and proposes itself as and cheaper alternative to laser trackers without sacrificing accuracy. Despite that, its main flaw lies on measurement noise, that despite being shown as almost negligible, can still affect tracking.

Figure 2.3: Optical CCM camera [2]

2.4 6DoF Tracking

6DoF, or six degrees of freedom, Tracking is known as the tracking of an object along not only the three-dimensional axis, but also its rotation along these axis, called roll, pitch and yaw.

The use of RGB-D cameras is common [14] [3], that is, cameras that determine the color and distance of tracked objects, like Kinect for example, along with probabilistic methods, like particle filters, based on three-dimensional renders of the tracked objects. These can be divided into two groups, whether they need previous information about the tracked object.

With no previous information, the systems need an alternative way to internalize which object is being tracked, and how it looks from a 3D perspective. This problem can be solved in a couple

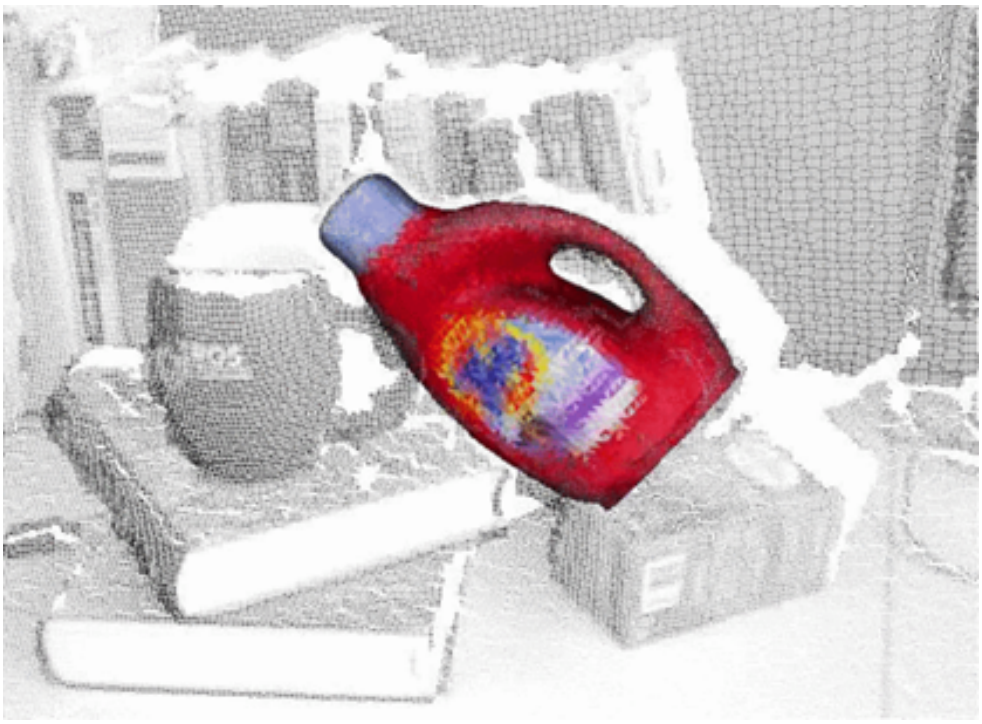

Figure 2.4: Tracking output from an RGB-D camera [3]

of ways, for example using the probability density of the object through Gaussian mixture models, or simply using object reconstruction programs, which allow for real time obtaining of an object's 3D model.

They both have disadvantages: the methods with no need for previous information are computationally heavy, or need extensive amounts of memory. On the other hand, methods that do need knowledge about the tracked object have difficulty in adapting to a more general use, needing to be adapted for each case. These methods are also susceptible to occlusion, that is, interferences in detection due to proximity of other objects other than the tracked one.

An alternative to RGB-D cameras is based on electromagnetic sensors, in particular in generating a bipolar electromagnetic field for tracking of the sensor placed on the object. This method is particularly useful due to not needing line-of-sight, making it robust against obstructions and occlusions.

Electromagnetic tracking systems usually consist on a field generator, responsible with magnetic field required to secure tracking, and a sensor. Given Faraday's law of induction, when a coil moves into a space where there's an alternating magnetic field, an electromotive force is induced in that coil, depending on the angle it makes with said field. Therefore, by measuring the voltage in that coil, it is possible to know the angle it makes with the magnetic field, and therefore one can learn the angle the coil itself makes with an axis. Give 3 coils placed perpendicularly to each other, the angles made with every axis are know given the full rotation of the object, and the intensity of the magnetic field provides the location.

The main disadvantages with this method lay with interferences due to nearby metallic objects, which can influence the magnetic field. Additionally, this method suffers a lot with range, as the farther you go from the generator, the weaker the field is.

Below is a table, including the price of some of the presented solutions, along with their frequency and price.

Table 2.1: Comparison between different tracking systems

Tests with prototype			
Name	Frequency	Error	Price
SteamVR 	60Hz	+-1mm	+-150 euros
Camera (Flex 3) 	60Hz	-	699$
Ultrasound (TR-DUO) 	1kHz	+-2cm	240 euros
Camara (Kinect) 	60Hz	+-1mm	200 euros

Analysing the table, it's easy to understand the benefits of the SteamVR system when compared with the other alternatives. This allows for absolute 6DoF pose tracking on a big area (10m x 10m), with low error and accessible price, mainly when compared with other alternatives.

Chapter 3

SteamVR

This chapter aims to provide an explanation on SteamVR, how it works, how it has evolved, and also about the system used for tracking, the SteamVR HDK.

3.1 Introduction

The HTC Vive was first unveiled during the Mobile World Congress on 1 March 2015. Developed by HTC and Valve Corporation, it is used as a virtual reality system, to be used mainly for video games alongside Steam, Valve's digital distribution software. With no licensing fees, hardware developers are open to produce devices and components to be used alongside the SteamVR system, allowing it to be applied to a multitude of projects.

3.2 SteamVR V1

SteamVR operates using two main components: base stations and photodiodes. Each V1 base station is composed by lasers able to make vertical and horizontal sweeps, plus IR (infrared) LEDs for periodic flashes.

The process starts with a flash from the LEDs, after which each photodiode starts counting the elapsed time until it gets hit by the laser sweep. With the laser rotation speed, it's possible to determine the photodiode's rotation around the yaw axis, in the case of the horizontal sweep, and the pitch axis, in the case of the vertical sweep. The objects know which of the base stations the flash is from by its length, which is also used to transmit other information like which rotor will start its rotation, and base station imperfections. The length of the signal relates to this information according to the table shown.

In reality, the base stations cannot generate perfect sweeps, due to small imperfections and non-ideal circumstances. These are accounted for after manufacturing, storing this calibration information on the base station, to be transmitted to the tracked objects via the LED flash.

Using the intersection of the sweep's measures, a line is obtained, being impossible to determine the exact location of the object with only one photodiode. For that, multiple photodiodes are

Figure 3.1: The inside of a SteamVR base station

needed, with the difference between the time intervals of the laser sweep allowing for the exact position of the object with two photodiodes. To know the orientation, one extra photodiode is needed, obtaining the 6DoF pose of the tracked object.

In truth, however, in order for SteamVR to successfully identify a pose from the tracked object, a total of four sensors need to be seen. On top of that, the sensors can't all be coplanar, so at least one of the sensors need to be on a different plane. Additionally, lower amounts of sensors may not allow for the whole object to be covered, which may lead to poor performance, as SteamVR doesn't use exclusively information obtained from sensors hit by the laser. If a sensor isn't hit by the laser at all, that also contributes to determining a pose, so lack of sensors on the opposite side of a face might lead to poor tracking.

SteamVR allows for errors consistently lower than 1mm thought a 5+ meter space. This superiority over camera-based systems is achieved thanks to two particular properties of SteamVR. First, the laser sweep of the room is analog, i.e. is based on a spinning motor, which allows for precise tracking along the room. Second and most importantly, with regular camera-based tracking systems, the farther an object is from the camera, the worse is the tracking, due to the limited resolution. In the case of SteamVR, the tracking quality is the same around the entire tracked space. This system also has all the information in the tracked object itself, instead of on a main computer, reducing the overall computational strain and improving reaction time.

Name	skip	data	axis	length (ticks)	length (µs)
j0	0	0	0	3000	62.5
k0	0	0	1	3500	72.9
j1	0	1	0	4000	83.3
k1	0	1	1	4500	93.8
j2	1	0	0	5000	104
k2	1	0	1	5500	115
j3	1	1	0	6000	125
k3	1	1	1	6500	135

Figure 3.2: Pulse length [4]

3.3 SteamVR V2

In 2018, Valve introduced a new version of the base stations, in order to reduce overall costs, as well as allowing the use of multiple base stations. This version has both the vertical and the horizontal sweeps being made using a single motor instead of two; however, the biggest change is based on the fact that the laser itself is coded with the angle and which base station it belongs to, making the initial flash obsolete and allowing for an easier coordination between multiple base stations.

3.4 Shoto HDK

The hardware used for tracking is called SteamVR Tracking HDK. It has four main components:

- Watchman Core Module: This module provides the processing power for the tracked object. It also contains the IMU.
- Application Board: Breaks the information processed by the Watchman Core module into ports that are more easily accessible, including an USB port.
- "Chiclet" Sensor Module: The small sensors that connect with the Breakout Board via flat cables.

- Sensor Breakout Board: Contains the connections for all 32 available sensors, 16 on each side.

The package also includes a dongle for wireless communication, an antenna and 32 flat cables to connect each sensor to the Breakout Board. This system has since been upgraded, and there's now a version that combines all three main modules into a single board, as well as combining the sensors into two 13-sensor high density connectors, for a total of 26 sensors.

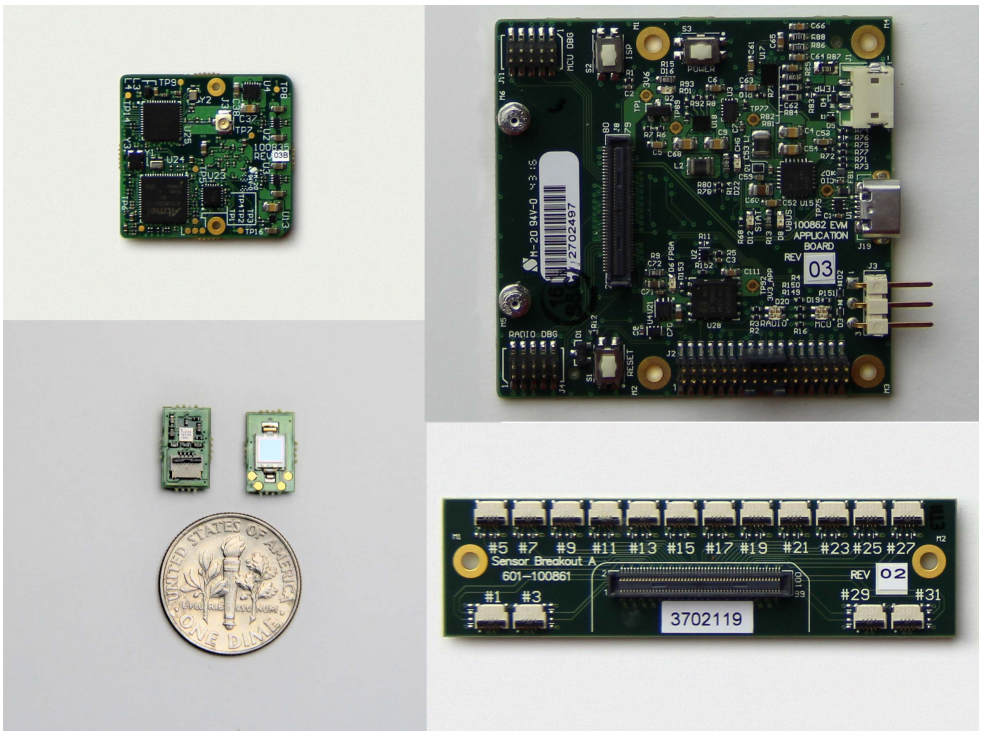

Figure 3.3: From top left to lower right: the Watchman Core Module, the Application Board, a "Chiclet" Sensor and the Sensor Breakout Board

Chapter 4

Setting up SteamVR

This chapter contains all relevant data needed in order to set up SteamVR from start to finish, starting with a completely unmodified object and making the necessary adjustments to obtain the best possible tracking on it.

4.1 3D Model

The first step in SteamVR tracking starts with the object to be tracked. While not obligatory, it is highly recommended to have a 3D model of the object, as this will make it easier to determine the position of the sensors in later steps. Additionally, should there be a need to display the object on SteamVR, a 3D model is necessary.

4.2 SteamVR Tracking HDK

The positions of the sensors is probably the most important factor in SteamVR. To help with this step, one program is vital: SteamVR Tracking HDK (not to be confused with the physical system of the same name being used to perform the tracking). This program can be accessed after obtaining a SteamVR Tracking License, after which it can be downloaded from the Steam launcher. This program contains many helpful tools, but the most helpful for sensor positioning is the HMD Designer GUI. This tool allows for determining the best sensor positions given a certain 3D model, as well as evaluating already positioned sensors.

After making a new project, the button "Add Input Files" is used to add the 3D models to the program, provided they have .stl or .scad extensions. After adding the object, the field "Model (.json) should be set to "Generate" and the field "Sensor Object (.stl or .scad) should be set to the name of your 3D model. Then some values can be tweaked in the "Simulate" section: "Total sensors" changes the total number of sensors placed (with a minimum of 5 and a maximum of 32). This should be set to whatever is appropriate to the project, keeping in mind that while more sensors allow for better tracking, above a certain number sensors stop adding any significant improvement to the tracking. "Simulation quality" has 5 possible options, with higher quality

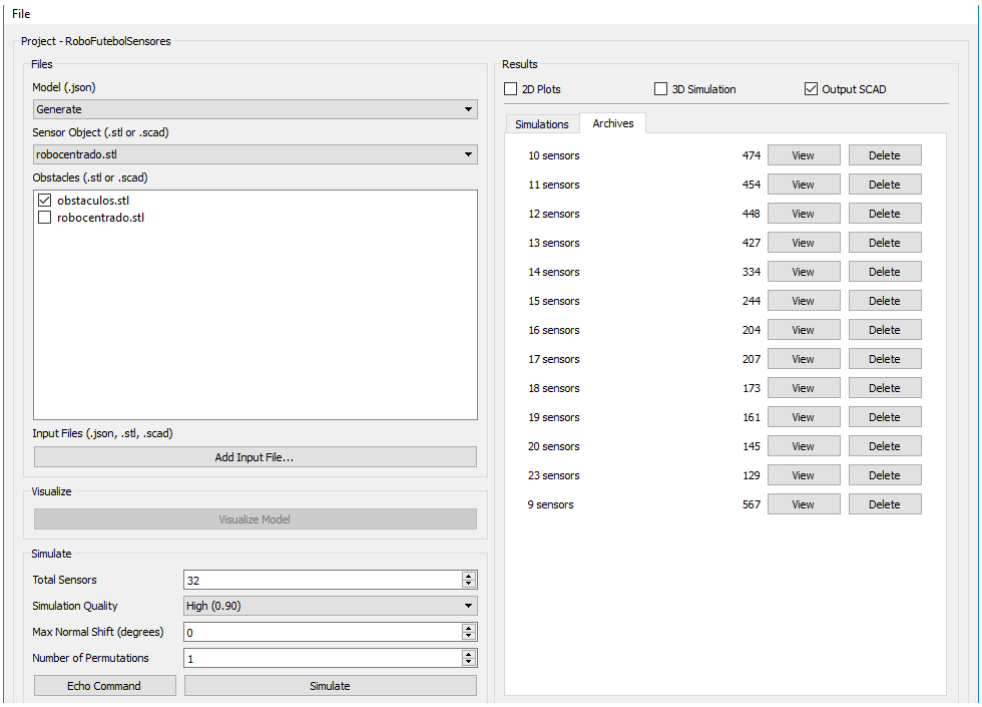

Figure 4.1: The HMD Designer project page

allowing for more accurate results. "Max Normal Shift (degrees)" is related to how much each sensor is allowed to rotate in the surface it rests on. Unless a sturdy method of fixing the sensors is used, or the object is too flat, 0 is recommended. "Number of Permutations" sets the number of simulations made. A higher number requires more computational power, but also a higher chance the most optimal positioning is reached. After everything is set accordingly, the button "Simulate" runs the program.

After the program finishes running, on the right side of the window all the executed simulations are shown. For each individual simulation, a score is given on a scale from 0 to 1000, with lower values meaning better tracking. There is also a "View" button and an "Archive" button. The "View" button shows three new windows, the most important being the Output SCAD. This window shows the position of the sensors determined by the program, as well as an editor specifying these positions in JSON. If the sensor disposition and quality is satisfactory, then the text displayed should be saved, especially the "modelNormals" and "modelPoints" field. The "Archive" buttons saves the simulation on the separate "Archives" tab, where it can be accessed at any time.

4.3 Calibrating the IMU

Next, the IMU of the system needs to be calibrated, in order to obtain precise measurements. This is done by connecting the system to a computer via an USB cable, and executing the "imu_calibrator" program. During this, the system should be turned along the six main orientations, pressing the "Enter" key in between each one. Th output at the end should be saved for later.

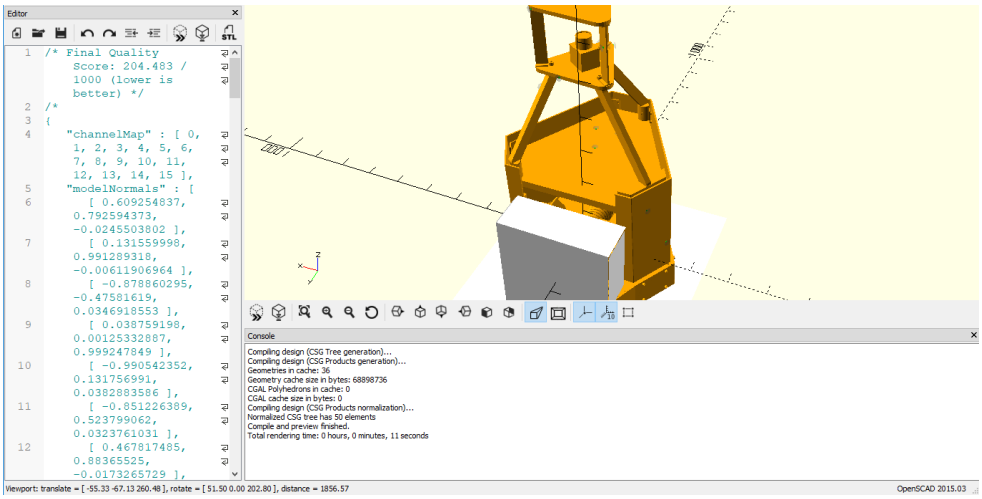

Figure 4.2: The Output SCAD window of the HMD Designer

4.4 The JSON file

With the position of the sensors determined, it's time to develop the JSON file. This is used to contain all information about the tracked object, including its type, the location of its sensors, which channels are being used, among others. The "the_json_file" document in the "docs" folder of SteamVR Tracking HDK details everything needed to make this file, but make sure to set the "modelNormals" and "modelPoints" field to what was saved from the HMD Designer, and the imu related fields from the results of "imu_calibrator". At the end of this step, a full, functional JSON file should now be available.

That is all in terms of files to set up. Next, the sensors should be distributed and properly fixed along the object with millimetrical precision, according to what was gotten from the HMD Designer. There should be special caution in making sure the sensors do not move, so as to prevent poor poses. The executable "lighthouse_console" should then be used (with the system connected to the PC via USB cable), in order to upload the JSON file. To do this, place this file in the same folder as "lighthouse_console", and use the command "uploadconfig [file name].json", with [file name] being the name of your JSON file. Be sure to power cycle the system (turn it off and back on), in order to apply the changes. By going back to "lighthouse_console" and using the "sesnorcheck" command, data exclusively from the sensors used should appear, including the number of base station hits for each sensor. This data can be used for debugging, by checking if each sensor is getting the expected number of hits.

4.5 Calibration

The last step before this object can be detected by SteamVR is calibrating the sensor position. SteamVR needs to know the precise location of the sensors, which may be slightly out of place, so this calibration is vital. This can be preformed using the "vrtrackingconfig" executable, though

it needs to be passed a specific parameter "/bodycal". This parameter is used to send the original, uncalibrated JSON, as well as two numbers: the total number of hits, and the number of hits per sensor required to complete the calibration. An example of a correct call to this program is "vrtracking.exe /bodycal [filename].json 800 200". Only one base station in mode A or B is needed to do the calibration. After pressing "Enter" when prompted, the object should be moved around with large, sweeping motions. For each successful pose, the program will print out a ":" until a sensor is fully calibrated, from which point it'll print out "." instead. A list of sensors that haven't finished calibrating is also shown, so the calibration only ends after the list is empty AND the max total hits have been achieved. After calibration, a file named "auto_[serial number].json" is left in the folder containing "vrtrackingcalib"; this file contains the calibrated points and should be uploaded to the system.

4.5.1 Working without the HMD

An optional step for this procedure is configuring SteamVR to work without an HMD. In this case we're only interested in the tracking part of SteamVR, having no need for a headset; however, by default, an headset is needed in order for SteamVR to work. This can be changed, however, by altering some settings on SteamVR. First, it's necessary to head to the Steam directory and follow the path "/steamapps/common/SteamVR/drivers/null/resources/settings" and open the "default.vrsettings" file in a text editing program. In here, the "enable" field should be changed to "true". This activates a null driver to take the place of the HMD, giving SteamVR something to consider the headset without physically having one. Back on the Steam directory, in the "config" folder, there's a "steamvr.vrsettings" file. This file should also be edited, below "steamvr", by adding or altering the following fields: "activateMultipleDrivers" : true, "forcedDriver" : "null" and "requireHmd" : false. After this, it should be possible to use SteamVR without an HMD.

With that, the object is fully calibrated and ready to use on SteamVR as a controller or a tracker.

Chapter 5

Results

This chapter contains the finds obtained through experimentation of the tracking systems, their comparison, and some conclusions to be taken from them.

5.1 Introduction

The Ground Truth used to evaluate the precision of the SteamVR system is Optitrack, consisting of 6 Flex 3 cameras and 4 reflective markers on the respective object. Each set of measurements was adapted using SVD, in order to obtain the transformations between both frames.

5.2 Setup used

In order to do the comparison of both systems and, conversely, determine the precision of SteamVR, the 6 Optitrack cameras were layed out in a circle, forming a work area of around two square meters. The SteamVR cameras were placed about 1.7 meters off the ground, in opposite ends of the area, with a distance between them of about 5 meters. The Optitrack system was calibrated and the origin point was made to be the middle of the work area.

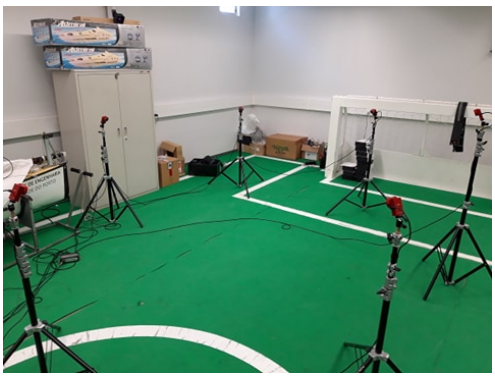

Figure 5.1: Setup incuding Optitrack cameras and one base station

Because both systems rely on IR, they were impossible to use simultaneously, thus limiting the means of testing the accuracy of SteamVR. This was solved by just placing the object in static position on the ground and measuring said position with each system individually, and then switching to the other. The results were saved and then, after five to seven measurements were made, they were then compared and the errors quantified.

5.3 Tests with prototype

For a first evaluation of the SteamVR system, a prototype was made that would be capable of providing tracking. For this step, a rectangular box was used, where attached, 2 on each of 3 vertex-sharing faces. The sensors were fixed with adhesive tape, and the HDK was held in place using rubber bands.

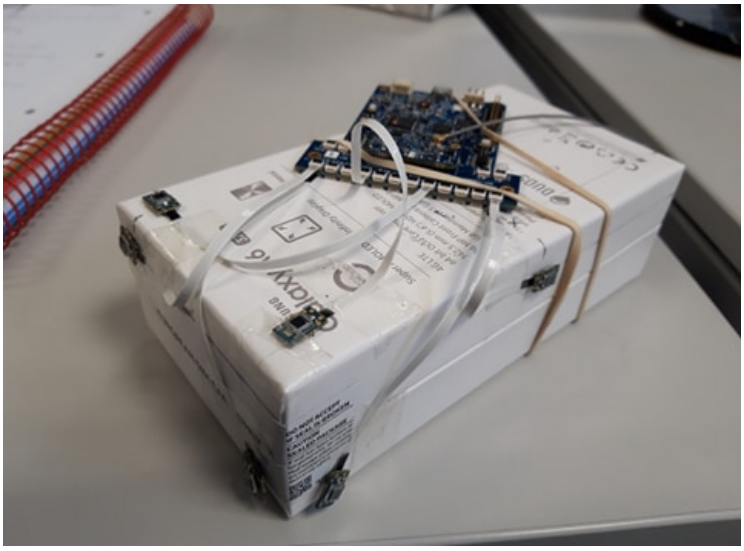

Figure 5.2: Prototype used for incial tests

Table 5.1: Tests with prototype

Tests with prototype				
Test	Error X(mm)	Error Y(mm)	Error Z(mm)	Error RMS(mm)
1	36,8	81,9	23,1	47,2
2	-9,1	-14,2	-38,0	-20,4
3	10,6	1,6	-22,9	-3,5
4	81,4	17,9	10,9	36,7
5	-16,4	-18,3	58,2	7,8
6	-66,5	13,0	-8,1	-20,5

5.4 Tests with controller

Noticing the magnitude of the errors obtained with the prototype, some tests with the controller from Vive were made, to determine whether the errors were inherent to the SteamVR system, or if they were just a consequence of the poor fixing of the sensors, or their small number. Thus, 4 Optitrack trackers were placed on the Vive controller so that it could be tracked by that system, being careful not to obstruct any SteamVR sensors with that placement.

Figure 5.3: Vive controller with Optitrack markers

Table 5.2: Tests with controller (1 base station)

Tests with controller				
Test	Error X(mm)	Error Y(mm)	Error Z(mm)	Error RMS(mm)
1	1,1	0,2	-0,4	0,7
2	-0,6	0,7	-2,6	1,6
3	-4,8	-0,6	2,5	3,1
4	-6,0	-1,0	2,6	3,8
5	4,5	2,5	-0,6	3,0
6	5,7	-1,8	-1,5	3,6

Table 5.3: Tests with controller (1 base station) - Angles

Tests with controller			
Test	Error Roll(°)	Error Pitch(°)	Error Yaw(°)
1	0.9	-0.8	-0.4
2	0.7	0.6	1.3
3	0.6	-1.1	-1.0
4	0.9	0.8	1.1
5	0.7	-1.0	-0.6
6	0.3	-1.2	-1.0

Given the small error obtained through these measures, it can be determined that the cause of the previous error was because of the rough placement and fixation of the sensors.

After these, 2 other scenarios where considered for testing, this time using both base stations. First, the application for mobile robots was considered, so tests where moving along the XY plane were performed, with little changes in the Z axis, or rotation other than yaw. Then, considering the application of industrial robots, tests were made with little movement but various different angles.

Table 5.4: Tests with controller (2 base stations) - Translation tests

Tests with controller (translation)				
Test	Error X(mm)	Error Y(mm)	Error Z(mm)	Error RMS(mm)
1	-8,5	-1,6	20,5	12,8
2	-2,8	6,5	9,7	6,9
3	2,8	-22,2	-11,2	14,5
4	-7,3	30,6	13,8	19,9
5	19,2	-20,4	-45,3	30,7
6	-3,3	7,0	12,5	8,5

Table 5.5: Tests with controller (2 base stations) - Rotation tests

Tests with controller (rotation)				
Test	Error X(mm)	Error Y(mm)	Error Z(mm)	Error RMS(mm)
1	14,6	10,1	-6,1	10,9
2	-10,1	6,0	-7,9	8,2
3	1,8	8,6	-9,3	7,4
4	8,9	2,7	-2,1	5,5
5	-15,3	-27,6	25,6	23,4

Table 5.6: Tests with controller (2 base stations) - Rotation tests (angles)

Tests with controller (rotation)			
Test	Error Roll(^{o})	Error Pitch(^{o})	Error Yaw(^{o})
1	-3.4	2.4	4.7
2	4.2	-1.7	-1.5
3	-2.3	-4.8	-3.9
4	-4.5	-3.9	2.4
5	1.8	3.6	-1.7

Curiously, readings provided by two base stations seemed more inaccurate than the ones given by one. However, some imperfections might have been at play, which might be able to explain these discrepancies. These imperfections are detailed on a later section.

5.5 Tests with support

As the final stage of testing, a support was 3D printed in order to make tracking on mobile robots possible. This support was made with the purpose of maintaining the sensors fixed in a stationary

position, solving the issues with the prototype. Additionally, it also contains a means to stick to the top of the robot, ensuring its tracking.

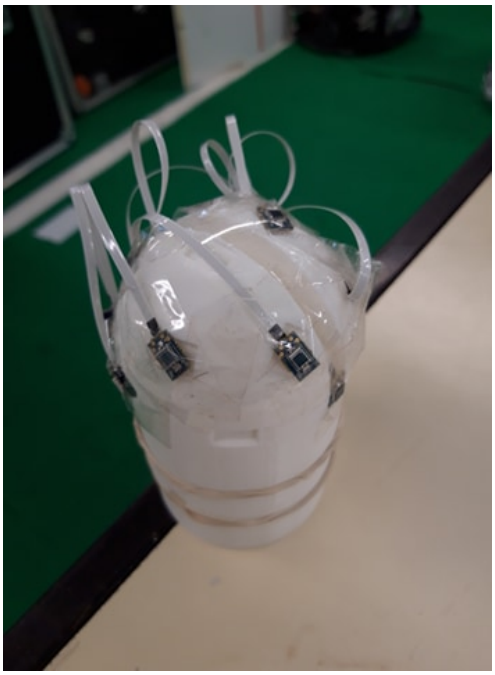

Figure 5.4: Support developed

The shape of the support is that of a cylinder with a half-sphere on top. Its has 64 millimeters of diameter and is 112 millimeters tall. On the semi-sphere part there are several indentations, of size 10 millimeters by 6 millimeters, being fourteen in total. These serve to house the sensors and prevent them from moving. There is also a big carving in the middle of the support, with the purpose of housing the HDK and keep it in place. A small hole passes through the cylinder in order to connect for the system to connect to the USB, and underneath a 13 millimeter orifice serves as the means of fixing it to the robot.

This support is capable of holding a total of 14 sensors; however, for testing, only 12 of the total were used.

Table 5.7: Tests with support (1 base station)

Tests with support				
Test	Error X(mm)	Error Y(mm)	Error Z(mm)	Error RMS(mm)
1	-6,9	20,6	-17,4	16,1
2	15,6	3,5	-33,5	21,4
3	29,7	-31,3	0,6	24,9
4	-34,4	48,8	20,1	36,4
5	-18,3	-10,8	7,3	13,0
6	31,7	-22,5	9,8	23,2
7	-73,8	-43,1	18,4	50,5

Table 5.8: Tests with support (2 base stations)

Tests with support				
Test	Error X(mm)	Error Y(mm)	Error Z(mm)	Error RMS(mm)
1	-16,0	18,1	-3,6	14,1
2	19,3	-36,9	37,9	32,5
3	-0,7	16,3	-33,3	21,4
4	-56,5	-13,5	-13,9	34,5
5	-32,3	-20,3	23,2	25,8
6	-27,8	22,9	-39,4	30,8

Due to the small size where the Optitrack system was setup, the testing was limited to a small space. In order to evaluate SteamVR's preformance on bigger areas, manual measurements were made.

Table 5.9: Tests with support (2 base stations) - Angles

Tests with support			
Test	Error Roll(º)	Error Pitch(º)	Error Yaw(º)
1	3,3	-3,5	-2,9
2	-3,3	-4,7	3,3
3	-1,0	-1,1	3,2
4	-2,9	-1,1	4,6
5	2,2	-2,4	1,9
6	4,0	-3,5	-2,4

Table 5.10: Tests with support (2 base stations) - Manual measures

Tests with support				
Test	Error X(mm)	Error Y(mm)	Error Z(mm)	Error RMS(mm)
1	-41,5	-175,4	-0,4	104,1
2	73,9	-50,3	9,6	51,9
3	-110,1	33,8	9,8	66,7
4	-281,9	37,9	1,8	164,3
5	33,6	53,7	8,8	36,9

5.6 Causes for discrepancies

Analysing the results obtained, while some might at first glance seem too inaccurate to consider, there were some persistent error sources that might have influenced the results of the tests.

- One pertinent error source is the discrepancy between the point that's being tracked by each tracking system. While if no rotation is preformed on the object this would have no

influence on the outcome, a simple discrepancy of a few millimetres between the tracked points of both systems can cause a big discrepancy on the readings. There was an effort to coincide these points as much as possible, but since this process must be done manually, there's room for some imprecisions.

- Regarding the Optitrack marker placement, given the small space available for placement, especially during the vive controller tests, some markers were placed slightly close to each other. This led to situations where the Optitrack system would sometimes merge two trackers into a single reading, which affected the quality of the tracking. This error is especially noticeable on table 5.4: the errors obtained in the vertical (Y) axis, mostly introduced by the Optitrack system, are strange considering the object was kept at ground level, and ignoring these significantly lowers the error RMS.

- Due to the small size of the sensors, they were difficult to properly attach to the support, since physical methods like 3D printing notches would be too small to be feasible, and temporary methods like tape would be too unreliable. Because of this, the SDK and the model needed to be calibrated often, which affected the measurements.

- Perhaps the biggest source of measurement errors refer to interferences between both systems. Namely, SteamVR sensors are also somewhat reflective, so they were being recognized by the Optitrack system as markers, which affected tracking.

- Conversely, there was also the risk of IR sweeps from the base stations reflecting off of the Optitrack markers and onto the SteamVR sensors, producing false measures and impacting tracking as well.

Chapter 6

Problems and Challenges

This chapter contains problems and challenges faced during the process of research, testing and writing of this thesis.

6.1 Lack of response from sensors

The first problem faced was an initial lack of response from sensors. Using the sensorcheck command in the lighthouse_console executable from SteamVR Tracking HDK, one can check on each individual sensor, verifying how many base station hits each one is getting, along with other data. This function is important to detect problems with individual sensors that may be affecting tracking. However, at first, this command returned no information whatsoever. After some research, the problem appeared to lie on the HDK's HWID (hardware ID). This is used to specify the hardware that is being used for tracking, as well as which sensors are being used. The HWID written to the development kit was referring to a different sensor than the ones being used, and so the information was being interpreted incorrectly. The HWID is usually set to 0x9003000[identifier], where the identifier field denotes what sensor is currently in use. The sensors used in this project are TS4231, whose identifier is 6, so the HWID was set to 0x90030006. After doing this, the output was being interpreted correctly, and the sensors started returning data correctly.

6.2 Flat object tracking

For a first evaluation of the system, a flat object was used to determine the first steps to take and to get accustomed to the system. 5 sensors were connected and disposed throughout a flat board , with one in each corner and one in the center. However, errors became immediately apparent in the sensor calibration phase: even with the individual sensors working correctly, the system failed to solve any poses of the object. At first, it was assumed the error came from the lack of precision on the positioning of the sensors, and to solve that issue a simple 3D base was printed to keep the sensors precisely in place. Despite that, the system continued to fail to solve for any poses. After some research, the true cause of the error was found: for a pose to be acknowledged by

SteamVR, a total of 4 sensors need to be seen, but at least one of them needs to be on a different plane than the other three. In the object, all five sensors were coplanar, and so no poses were being solved. To counter this issue, a new arrangement of the sensors was considered, where this time 6 sensors were used, 2 on each face of a rectangular box, for a total of 3 adjacent faces. With this distribution, calibration was successfully executed, and the object started being recognized in SteamVR.

6.3 Errors during calibration

At some points during calibration, a error would occur, with the console printing "Unexpected error 9?". This error seemed to be related with multiple base stations being detected while calibrating, however only one was being used. At first the error was attributed to interference from some close-by source, but it was then discovered that the error lied in positioning the object too close to the base station, which made it think there were multiple base stations in range when in reality there was only one. This error stopped occurring after maintaining a minimal distance to the base station of one meter.

6.4 Consistent errors

As progress was made, the box used to first test the SteamVR tracking and begin setting up the process began to become obsolete. The error on the placement of the sensors was too high due to them being kept in place through tape, and the number and disposition of the sensors only allowed the object to be detected on certain poses facing the base station. Because of that, the error given when compared to the Optitrack ground truth would sometimes be above 100mm, which was much too high especially with what was expected. Thus, a new support was made, in order to keep the sensors firmly in place, as well as allowing detection of the object in various poses around it.

6.5 Sensor reflecting

As referred on the last chapter, both tracking systems used, SteamVR and Optitrack, often interfered with each other, making it difficult to obtain precise measures. This was especially noticeable while testing with the support: due to the relatively large amounts of sensors used, the interference made on Optitrack system, due to the IR rays reflecting of the SteamVR sensors, made detecting markers on the support practically impossible. Solving this problem required fixing the support to a box and placing the markers there, farther away from the support. This enabled the markers to be properly detected and made it possible for Optitrack to output precise measurements.

Chapter 7

Conclusion and Future Work

7.1 Analysis

Starting with the prototype, the error magnitude observed was too high to be used for the desired purposes. The biggest error on table 5.1 almost reaches 90 millimeters, which, given that the range expected was around 1 millimeter, would heavily impact its usage. However, since the number of sensors on the object was low, and they weren't very firmly fixed to it, the error was justified, if not expected. Regardless, it served to establish sensor position as a important factor in SteamVR tracking.

As mentioned, the tests with the Vive controller had as the main objective to determine the source of the errors obtained with the prototype, as well as set a base on the actual error to be expected from the SteamVR system. The first tests, with one base station, were closer to what was expected, given the small errors, with the biggest of which being 5 millimeters. These tests confirmed the possibility of SteamVR providing accurate tracking even with just one base station, despite the area considered not being very big.

Following tests using 2 base stations gave bigger errors, however. In some instances it reached 30 millimeters, which while not as bad as the prototype measurements, certainly isn't ideal. Despite that, there is evidence that those results where influenced by some interference, namely the odd amounts of error in the Y axis. Given that the objects were always placed on the ground, big errors in the vertical axis are suspicious. Coupled with the fact that the Optitrack system seemed to be at fault for those amounts of errors seemed to indicate the measurements were influenced by the close position of the markers on the Vive controller.

Lastly, the tests on the support were preformed. The readings were similar to the ones given by the tests with the Vive controller, if slightly worse. Due to the big amount of SteamVR sensors, the interference was big, which certainly played a big part on the magnitude of the errors.

The rough tests using the manual measurements can only be used to give a hint as to the performance of the system in bigger areas. The errors were expectedly big, since precision utilizing real world measurements is poor. However, the system still seemed to be able to detect the position with accuracy.

Table 7.1: Readings on the Y axis by both systems

Readings on the Y axis by both systems		
Test	Y (SteamVR)	Y (Optitrack)
1	0.0553	0.076304
2	0.0562	0.105311
3	0.0533	0.04216
4	0.0512	0.080082
5	0.0553	0.042492
6	0.0599	0.102391
STD	0.001584452	0.014192553

Given the results obtained, it can be concluded that use of the SteamVR system in the mobile robotics field is realistic and appropriate. The errors shown were small enough that they justify the smaller price and number of base stations. Given that the precision with one or two base stations seems to be comparable, even simply one might be able to provide accurate tracking for this purpose.

Conversely, for the field of industrial robotics, the results leave much to be desired. Given the fact that interferences might have been at play during tests, there is a possibility that the SteamVR system might provide errors small enough that it could be considered, but the precision given by the test rarely dropped below 1 millimeter, it is unlikely that the SteamVR system possesses the precision to be used in such a delicate application.

7.2 Objective satisfaction

In general, the main objectives of this thesis were achieved: a SteamVR tracking system was successfully implemented as well as a means to use it for robot tracking. Its uses for both mobile and industrial robotics were studied, and a conclusion was made on this regard. A support was developed, sporting optimized sensor locations and ability to be mounted on firmly on football robots.

While this project aimed to look into a solution for the small size of the cables in the SteamVR kit, this work turned out to be unnecessary, since the sensors were not spread around the robot but contained to a small space on top of it. Given that the spread of the sensors doesn't seem to affect tracking, sensors that are far apart give roughly the same amount of error as close-by ones, and so that possibility remained unexplored.

7.3 Future Work

Some future tests could be done in order to determine the efficiency of the SteamVR system given more than one tracked object. Because SteamVR base stations are passive emitters there should be no influence on the tracking precision, but there is still possibility for interferences.

Additionally, given an accurate tracking system that doesn't use IR, more tests could be done in order to accurately and precisely determine the error of SteamVR. Finally, all the tests preformed had the tracked object connected to the PC, but there's the possibility that wireless tracking might influence the accuracy of the readings.

Appendix A

The JSON file

This appendix aims to share the used JSON file as well as explain its creation process and what each field represents.

A.1 Example of a JSON file

Presented here is the JSON file used in the case of the support developed.


```
{
    "device_class" : "controller",
    "device_pid" : 8960,
    "device_serial_number" : "LHR-FCA3AF36",
    "device_vid" : 10462,
    "firmware_config" : {
        "mode" : "controller",
        "radio" : true,
        "sensor_env_on_pin_a" : 0,
        "spi_flash" : true,
        "trackpad" : true,
        "trigger" : true,
        "vrc" : true
    },
    "head" : {
        "plus_x" : [ 1, 0, 0 ],
        "plus_z" : [ 0, 0, 1 ],
        "position" : [ 0, 0, 0 ]
    },
    "imu" : {
        "acc_bias" : [ 0.109899998, 0.0903600007, -0.139400005 ],
        "acc_scale" : [ 0.99940002, 0.997300029, 0.99059999 ],
```

```
        "gyro_bias" : [ -0.0427699983, 0.00515800016, 0.00669099996 ],
        "plus_x" : [ -1.054362725e-07, -3.604019226e-08, 1.0000292 ],
        "plus_z" : [ 0.63439340829, 0.77301038931, 4.6964919064e-08 ],
        "position" : [ 0.01837852969, 0.02611438371, -0.02289948239 ]
    },
    "lighthouse_config" : {
        "channelMap" : [ 7, 19, 5, 27, 23, 26, 22, 24, 1, 4 ],
        "modelNormals" : [
            [ 0.416179389, 0.778616667, 0.469628304 ],
            [ -0.416178912, 0.778616011, 0.469629914 ],
            [ 0.560082912, -0.68246299, 0.469629019 ],
            [ -0.956940234, 0.290285081, 2.45697663e-07 ],
            [ 0.634393394, 0.773010433, 7.37949932e-08 ],
            [ -0.770780504, -0.0759151429, 0.632561684 ],
            [ -0.365101159, -0.683056653, 0.63256216 ],
            [ 0.0980166867, -0.995184779, -1.14073885e-07 ],
            [ 0.844848394, 0.256281912, 0.469628423 ],
            [ 0.452793717, -0.137353361, 0.880972207 ]
        ],
        "modelPoints" : [
            [ 0.0126897674, 0.01967833, 0.0421091057 ],
            [ -0.00901251659, 0.0213944688, 0.042144537 ],
            [ 0.0177290048, -0.0185603052, 0.037962988 ],
            [ -0.0259952229, 0.0105418535, 0.023831306 ],
            [ 0.0159164928, 0.0231517218, 0.0232310984 ],
            [ -0.0233178828, -0.00341264927, 0.0425658748 ],
            [ -0.0087489821, -0.022751892, 0.0423804559 ],
            [ 0.00129639416, -0.0276171342, 0.024099743 ],
            [ 0.0251254495, 0.00508707855, 0.0381087512 ],
            [ 0.0106927138, -0.00260362588, 0.0529645421 ]
        ]
    },
    "render_model" : "teste",
    "revision" : 4
}
```


A.2 Header

The *"device_class"* determines what type of device is being used: either a controller or an HMD. This is used to determine whether to display an object in VR or to associate it with a display; in

this case, the former is used. The following 3 fields are particular to the device's manufacturer, though the values attributed to *"device_vid"* and *"device_pid"* are used for prototyping purposes.

A.3 Head

This field has a different meaning whether the device is a controller or an HMD. For a controller, it serves to define the relation of the orientation of the 3D model with the tracked object. This section is unimportant for the use in study, so it remains with its default values.

A.4 IMU

This section is used to store IMU calibration data. The first 3 fields relate to the bias and scale errors the IMU might have, and are obtained through the imu_calibrator tool. These are used to make up for any imperfections the IMU might have, so the tracking results can converge faster. The *"plus_x"* and *"plus_z"* fields are used to set the orientation of the X and Z axis of the IMU through the objects coordinate system; a right-handed approach is considered in determining the subsequent orientation of the Y axis. Finally, the *"position"* simply specifies the location of the IMU on the device.

A.5 Lighthouse_config

The last field pertains to the sensors, their positions and orientations. The *"channelMap"* lists the sensor channels used, and the order described here is maintained through the next two fields, so the channel described first corresponds to the first entry, regardless of being a higher number or not. The *"modelNormals"* field describes the orientation of the sensors through their normal vectors. These should refer to an unit vector. *"modelPoints"* simply contains the positions of the sensors, in meters. The *"render_model"* section pertains to the display in VR of the tracked object: to be displayed, an object must first be given a model, which must be saved to the SteamVR folder "rendermodels", located in

`\steamapps\common\SteamVR\resources\rendermodels`

. Since VR displays are not the focus of this thesis, a very basic model was provided and saved to said folder.

References

[1] Inside kinect controller. Disponível em https://www.researchgate.net/figure/Inside-Kinect-controller-134_fig5_303974799, acedido a última vez em 12 de Junho de 2019.

[2] S. Gharaaty, Tingting Shu, Wen-Fang Xie, A. Joubair, and I.A. Bonev. Accuracy enhancement of industrial robots by on-line pose correction. *2017 2nd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*, 2017. URL: https://www.engineeringvillage.com/share/document.url?mid=inspec{_}M78f14f1a15dcda54850M496e10178163176{&}database=ins, doi:10.1109/ACIRS.2017.7986096.

[3] Changhyun Choi and H.I. Christensen. RGB-D object tracking: a particle filter approach on GPU. *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013)*, 2013. URL: https://www.engineeringvillage.com/share/document.url?mid=inspec{_}M17280392143e4050e0eM68152061377553{&}database=ins, doi:10.1109/IROS.2013.6696485.

[4] Pulse length. Disponível em https://github.com/nairol/LighthouseRedox/blob/master/docs/Light%20Emissions.md, acedido a última vez em 12 de Junho de 2019.

[5] F. Blais, J.-A. Beraldin, S.F. El-Hakim, and L. Cournoyer. Real-time geometrical tracking and pose estimation using laser triangulation and photogrammetry. *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, 2001. URL: https://www.engineeringvillage.com/share/document.url?mid=inspec{_}base906978229{&}database=ins, doi:10.1109/IM.2001.924436.

[6] M. Veloso, P. Stone, and Kwun Han. The CMUnited-97 robotic soccer team: perception and multi-agent control. *Robotics and Autonomous Systems*, 29(2-3), nov 1999. URL: https://www.engineeringvillage.com/share/document.url?mid=inspec{_}base906447973{&}database=ins, doi:10.1016/S0921-8890(99)00048-2.

[7] A.C. Stephon and S. Khorbotly. A camera-based target tracking system for football playing robots. *2012 Southeastern Symposium on System Theory*, 2012. URL: https://www.engineeringvillage.com/share/document.url?mid=inspec{_}10655dd13799829e17M70192061377553{&}database=ins, doi:10.1109/SSST.2012.6195114.

[8] How it works: Xbox kinect. Disponível em https://www.jameco.com/jameco/workshop/howitworks/xboxkinect.html, acedido a última vez em 21 de Junho de 2019.

[9] J.T. Kostamovaara, A.J. Makynen, and R.A. Myllyla. Method for industrial robot tracking and navigation based on time-of-flight laser rangefinding and the position sensitive detection technique. *Proceedings of the SPIE - The International Society for Optical Engineering*, 1010, 1989. URL: https://www.engineeringvillage.com/share/document.url?mid=inspec{_}base803454529{&}database=ins, doi:10.1117/12.949221.

[10] Chung-Hao Chen, Chang Cheng, D. Page, A. Koschan, and Mongi Abidi. Tracking a moving object with real-time obstacle avoidance. *Industrial Robot*, 33(6), 2006. URL: https://www.engineeringvillage.com/share/document.url?mid=inspec{_}13f30451102b9e2c38M6f702061377553{&}database=ins, doi:10.1108/01439910610705635.

[11] M. Ferreira, P. Costa, L. Rocha, and A.P. Moreira. Stereo-based real-time 6-DoF work tool tracking for robot programing by demonstration. *International Journal of Advanced Manufacturing Technology*, 85(1-4), jul 2016. URL: https://www.engineeringvillage.com/share/document.url?mid=inspec{_}49433b2415807bd9145M6c9f10178163171{&}database=ins, doi:10.1007/s00170-014-6026-x.

[12] A. Nubiola and I.A. Bonev. Absolute calibration of an ABB IRB 1600 robot using a laser tracker. *Robotics and Computer-Integrated Manufacturing*, 29(1), feb 2013. URL: https://www.engineeringvillage.com/share/document.url?mid=inspec{_}10655dd13a4644e835M54752061377553{&}database=ins, doi:10.1016/j.rcim.2012.06.004.

[13] V. Lertpiriyasuwat and M.C. Berg. Adaptive real-time estimation of end-effector position and orientation using precise measurements of end-effector position. *IEEE/ASME Transactions on Mechatronics*, 11(3), jun 2006. URL: https://www.engineeringvillage.com/share/document.url?mid=inspec{_}13f304510c7ced49cbM7b612061377553{&}database=ins, doi:10.1109/TMECH.2006.876515.

[14] S. Akkaladevi, M. Ankerl, C. Heindl, and A. Pichler. Tracking multiple rigid symmetric and non-symmetric objects in real-time using depth data. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016. URL: https://www.engineeringvillage.com/share/document.url?mid=inspec{_}M22a3540615579506d43M494210178163171{&}database=ins, doi:10.1109/ICRA.2016.7487784.