Imperial College London

Department of Electrical and Electronic Engineering

Final Year Project Report 2019

Project Title:	**Augmented Reality-assisted Human Robotic Interaction**
Student:	**Aufar P. Laksana**
CID:	**01093575**
Course:	**EIE4**
Project Supervisor:	**Dr Yiannis Demiris**
Second Marker:	**Dr Tae-Kyun Kim**

Abstract Powered wheelchairs are becoming increasingly commonplace in the modern world. However, a major issue faced by powered wheelchair users (PWUs) is navigating the device in crowded areas. Controlling the powered wheelchair in crowded areas requires increased concentration from the PWU, as people in crowds often move unpredictably, or are hidden from view due to standing behind another person or object.

This project utilizes computer vision techniques to predict the direction of travel of individuals in crowds, and implements an augmented reality system using the Microsoft Hololens that aids the PWU by displaying visual aids that indicate the motion of people. The system further aids the user by warning the user of potential collisions, allowing the PWU to make better navigation decisions. The project also explores the use of the system as a method of assitive control of the wheelchair, preventing collisions by stopping the powered wheelchair should the PWU not notice an individual crossing their path.

Contents

Chapter 1

Introduction and Requirements

1.1 Introduction

This report was written as part of the Final Year Project for the MEng Electronic & Information Engineering course. The project was supervised by Dr. Yiannis Demiris at the Imperial College London.

1.2 Motivation

Chapter 2

Background

This project is focused on computer vision for detecting and tracking humans in the surroundings, estimating their trajectories and distance from the PWU, the reactive control systems that prevent collisions with the detected objects as well as the augmented reality display to provide visual cues to the PWU.

2.1 Human Detection

Human detection is a subset of the classic computer vision problem of object detection. In order to develop an augmented reality system that will help PWUs to navigate in public spaces, it is essential for the system to be able to discern humans from the surroundings.

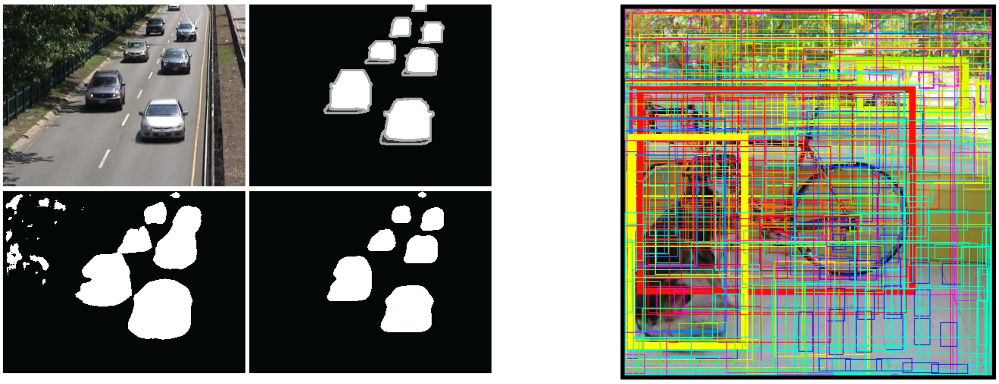
2.1.1 Direction of Research

The problem arises in crowded areas, whereby individuals are occluded by other people or objects in front of them, leaving only certain body parts visible. As such, we began our research with the problem of being able to detect people in images where identifiying parts of the body are not always visible.

2.1.2 Review of Existing Methodologies

A related field of research is that of people counting and human detection in visual surveillance in public areas. Where the problem differs is that surveillance benefits from being able to rely on cameras with a good view of the crowd from above, whereas for a PWU, the camera will not have as high of a vantage point, making detecting every single individual in a crowd impossible.

Despite the disadvantage, similar techniques can be used to detect humans in video. Most methods can be classified into two categories [1]. The first technique, foreground detection, attempts to model the background of an image and then detect the changes that occur between frames. The second category involves exhaustively searching the image with a scanning window, and deciding if each window can be classified into a human shape.



(a) Foreground Detection [2] (b) Scanning Window [3]

Figure 2.1: Comparison of Foreground Detection and Scanning Windows

Foreground Detection

Background subtraction is a widely used approach for detecting moving objects [4]. A temporal average filter can be used to find the median of all the pixels in an image to form a reference image. Frames with moving objects can then be compared pixelwise to the reference, and a threshold set to determines if the pixel is part of the background or foreground. People counting and human detection can then be achieved by segmenting the foreground image into individuals.

However, this technique often relies on a static camera in a well placed location. This brings up several reasons as to why this method would not be suitable for this project. Firstly, the camera available is part of a head-mounted augmented reality device. The wearer has the ability to move the camera in 6 degrees of freedom. Secondly, the wearer will also be navigating a powered wheelchair. As a result, the background is constantly changing, and the reference image would require constant recomputation before human detection can even begin.

Scanning Windows

Due to the ever-changing surroundings of a mobile robot, a better approach for object detection is to exhaustively search an image using scanning windows and determining if an object was detected in each window. However, it must be noted that this method is computationally expensive. In order to achieve real-time detection on a mobile robot, the use of a graphics processing unit (GPU) should be considered [5].

Classical Object Detection

Haar Cascades Haar cascades classifies images based on the value of simple features [6], which are variants of the difference between the sum of pixel values in rectangular regions. An intermediate representation of the original image is used to rapidly compute a small set of representative rectangular features.

A cascade of classifiers is then used to determine if the region is detected as a human. The detection process is that of a degenerate decision tree, where a positive result in the first cascade will trigerr an evaluation in the second, more successful classifier. As such, the initial classifier can eliminate a large number of negative examples with very little processing. After several stages, the number of sub-windows has been reduced radically

Histograms of Oriented Gradients The method proposed is implemented by dividing the image window into small spatial regions and calculating a local 1-D histogram of gradient directions for all the pixels in the region. The combined local histograms form the overall feature representation of the image.

The detection window is tiled with the Histogram of Oriented Gradient (HOG) descriptors. In the original paper [7], these feature vectors were then used in a conventional SVM based window classifier to give human detections.

Deep Learning Object Detection

Modern approaches for human detection largely depend on Deep Convolutional Neural Networks (CNN). The approach provides the best in class performance, as well as scaling effectively with more data. An added advantage of using CNN based object detection systems for this project is that they are also capable of detecting multiple classes of objects.

An issue with CNN approaches is that the methods are trying to draw bounding boxes around objects of interest in images. However, we do not know the number of objects in the image beforehand. As such, to be completely sure every object has been detected, a naive solution is to take a huge number of regions and attempt to classify all the objects in the region, a computationaly expensive process.

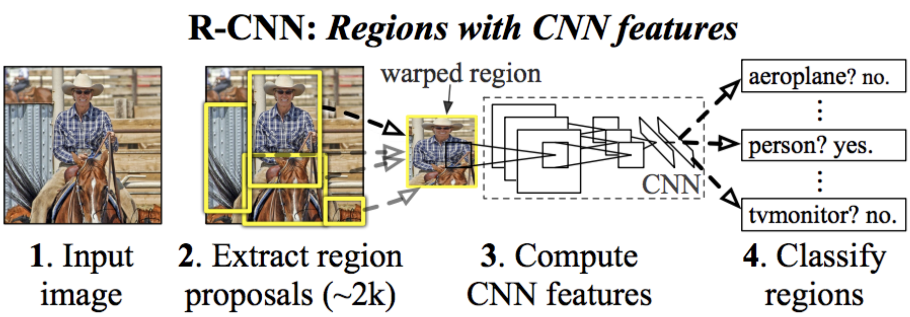


Figure 2.2: R-CNN Approach

R-CNN The R-CNN method uses a selective search to extract 2000 regions from an image [8]. The regions are selected by generating a large number of candidate regions and using a greedy algorithm to recursively combine similar regions into larger ones. The regions are then fed into a CNN that acts as a feature extractor and the output dense layer consists of the features extracted from the image, which are then fed into an SVM to classify the presence of objects in the region.

The major disadvantage to this approach is the amount of time required to train the network. Each training image has to be classied once for each of the 2000 region proposals. Furthermore, the selective search algorithm is a fixed algorithm (no learning is done), and as such, could lead to generation of bad candidate region proposals.

YOLO Whereas R-CNN uses regions to localize the object within an image, You Only Look Once (YOLO) looks at the image as a whole and uses a single CNN to predict the bounding box and the class probabilities [3]. By looking at the image as a whole, the network can use features from the entire image to predict each bounding box.

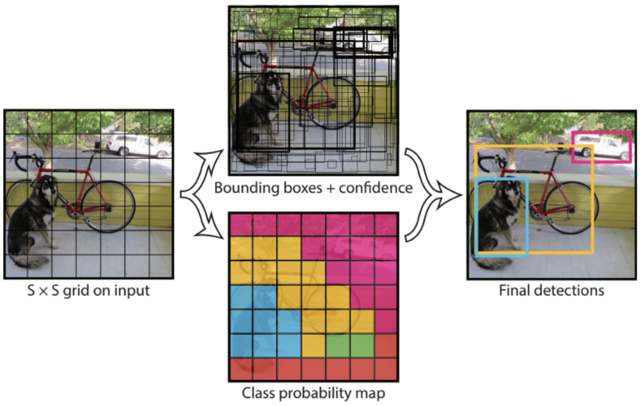


Figure 2.3: YOLO Approach

The model divides the image into an $S \times S$ grid, and for each cell, predicts a number of bounding boxes, the confidence for those boxes and the class probabilities.

2.1.3 Comments

As seen from the research, we can clearly see that there are many ways to solve the human detection problem. The classical approaches, although computationally efficient, are siginficiantly outperformed by the deep learning approaches. For a mobile robot in a public area, we want to be able to detect almost all humans in the surroundings to better inform the PWU.

However, the major disadvantage of the deep learning approach is the time taken to train the network, as well as the requirement of a GPU to achieve real-time performance. These issues will be addressed in a later section of the report.

2.2 Object Tracking

Object tracking can be defined as the ability to detect objects in consecutive frames and determining if the same objects are present. The techniques are often used in security and surveillance to track individuals across multiple cameras. A more relevant use of object tracking is in augmented reality with ARMarkers to allow for more accurate placements of holograms as the user moves through the AR world.

2.2.1 Direction of Research

A common scenario for PWU in public spaces is having multiple people walking in the surroundings. Ideally, the augmented reality system should be able to track the same people across frames to be able to determine their direction of motion. As such, we focus our research on the multiple object tracking (MOT) problem in real-time. For an augmented reality system for a PWU, the object tracking must be done in real-time in order to feedback to the PWU. This narrows our field of research to online object tracking techniques.

2.2.2 Review of Exisiting Methodologies

Pedestrian detection is often achieved by using a high quality object detector and associating the detections across frames [9]. The associations are based on the appearance and location similarity. Furthermore, it is possible to discern simple motion patterns from the tracked pedestrians, allowing for more accurate tracking.

SORT

Methodology The Simple Online and Realtime Tracking (SORT) method relies on the accurate detections of a CNN to calculate bounding boxes of the tracked objects across frames [10]. The technique estimates the inter-frame displacements of each detected objects with a linear constant velocity model. The state of each tracked object is modelled using the bounding box centroids u and v, the scale and aspect ratio, s and r.

$$x = [u, v, s, r, \dot{u}, \dot{v}, \dot{s}]$$

When a new detection is associated with a tracked object, the bounding box of the new detection is used to update the tracked object state, and using a Kalman filter to update the velocity components [11]. To determine associations between new detections and tracked targets, the SORT algorithm relies on the intersection-over-union (IOU) distance between each detection and the predicted bounding boxes of all the existing targets.

For every detection to be tracked, a uniqe tracker identity must be created and destroyed when the object enters and leaves the image. The original implmentation of the algorithm relied on a IOU_{min} value to signify the existence of an untracked object. The tracks are then terminated if they are not detected for an allotted number of frames, to prevent the unbounded growth of trackers.

Limitations Due to the simplicity of the association metric, the significant overhead and complexity of object re-identification is removed, allowing for the system to work in real-time applications. However, this also reduces the accuracy of the tracking, since occlusions will spawn new trackers for the same objects. Furthermore, the accuracy of the tracking is largely dependent on the object detector providing accurate bounding boxes.

Deep SORT

The original SORT suffered from a high number of identity switches, since the association metric was only accurate if the state estimation uncertainty was low. Wokje proposed a solution to the issue by learning a deep association metric on a re-identification dataset [12].

Methodology The tracking and Kalman filtering in Deep SORT is mostly identical to the original SORT implementation. However, Deep SORT uses a Mahalonibis distance as an association metric between the Kalman predicted states and new detections. It further uses a second metric, whereby an appearance descriptor is calculated for each bounding box. A gallery of the previous $L_k = 100$ descriptors are kept for each track. The algorithm then iterates and measures the smallest cosine distance between the existing tracks and the detection.

The appearance descriptor is implemented using a CNN that has been trained offline on a person re-identification dataset. The Github implementation of the Deep SORT algorithm uses a simple nearest neighbour query without any additional metric learning.

Limitations Although the accuracy of the the tracking is improved and the issue of occlusions is reduced, the increased complexity of the algorithm requires more computational power. As stated in the paper, a modern GPU would be required to run this in real-time, due to the need for an appearance descriptor to be calculated for each detection.

2.2.3 Comments

For this project, we have limited ourselves to researching simple object tracking methods that work in real-time. We can clearly see a trade-off between accuracy of tracking and computational power. Further investigation into the hardware available and the importance of object tracker accuracy will be needed to decide what method would be best for the augmented reality system.

2.3 Head and Body Pose Estimation

Pose estimation is a general computer vision problem where we attempt to detect the position and orientation of an object. This process can be achieved by detecting keypoint locations that describe the pose of the object. For instance, in body pose estimation, we identify the joints in the body.

2.3.1 Direction of Research

An interesting concept to explore is that of head and body pose estimation as a way of infering the direction a person is walking in. For instance, people tend to look in the direction they are currently walking, but should they want to change direction, they also tend to look in that direction before changing [13]. Similarly, if we can determine the body pose of a person, the system will be able to tell if a person is walking to or away from the PWU without relying on depth sensors.

2.3.2 Review of Existing Methodologies

Head Pose Estimation

Head pose estimation is intrinsically linked with visual gaze estimation [14]. If we can characterize the direction and focus of a person's eyes, it may be possible to determine the direction they will walk in next.

Facial Landmark Detection Before head pose estimation can be done, keypoints on the face must be detected [15]. These points will then be used to solve a Perspective-n-Point (PnP) problem to determine the head pose. There are many facial landmark detection techniques, depending on the number of landmarks to be detected. As the number of landmarks increase, the more accurate the pose estimation can be. However, it also increases the complexity of the detection, and as such, it becomes a trade-off between the two factors.

Body Pose Estimation

An idea we wish to explore is using the body pose of an individual to estimate the direction they are walking in. If the system can discern between a person's back or front, we can infer the motion, since people do not normally walk backwards. A limitation of our system is that it has to be done in real time for it to be effective. As such, the techniques we can explore are limited by the hardware available.

PoseNet A common approach for body pose estimation is to employ a person detector and perform single-person pose estimation for each detection, known as a top-down approach. PoseNet is a real-time human pose estimator with a web-browser implementation that runs on Tensorflow.js, making it easily available to anyone. The implementation is based on the works of Papanderou and Zhu [16] in building a network that utilizes the Faster-RCNN model as an object detector to obtain accurate bounding boxes of people in an image [17]. The keypoints are then calculated using a a ResNet [18] by predicting activation heatmaps and offsets.

OpenPose Top-down approaches can be limited by the failure of the person detector. This is especially common when two individuals are very close to each other, and the detector is unable to differentiate between them. In contrast the bottom-up approach, which is based on partitioning and labeling an initial pool of body part candidates into subsets [19], is able to deal with an unknown number of people, and can infer that number by linking the part hypotheses.

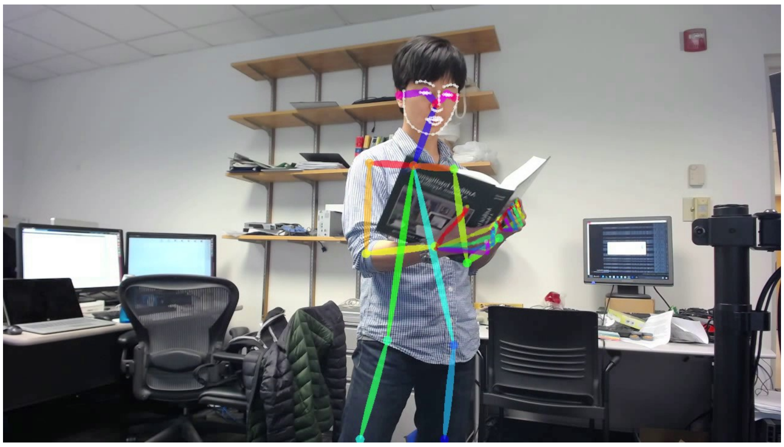


Figure 2.4: OpenPose body pose detection [20]

As such, OpenPose presents a method of multi-person pose esimtation using a bottom-up approach [20]. The method relies on *partial affinity fields* (PAFs), a representation that encodes unstructured pairwise relationships between body parts. The network produces the 2D confidence maps of body part locations and PAFs, and through greedy inference, the network outputs the 2D keypoints for all people in the image.

2.3.3 Comments

Head and body pose estimation is a vast field of research, with dozens of effective real-time estimation methods. This project is not focused on achieving the best body/head pose estimation, but rather, in utilizing existing frameworks to infer directions of individuals. As such, we have refrained from delving too deep into the theory of body pose estimation, and instead, have attempted to choose a method from available implementations.

2.4 SLAM

The term mapping refers to a system that will create a map of the surrounding areas, by detecting objects such as walls and other obstacles. In order to help users navigate, the system must analyse the surroundings for potential dangers. As such, it is important to build up a thorough and complete map.

A fundamental method for robot navigation is the Simultaneous Localization And Mapping (SLAM) method. The process allows the system to predict the trajectory of the robot and the location of all objects on-line, without the need of an *a priori* knowledge of the robots location [21]. The method estimates the pose of the robot relative to landmarks which are detected. The popularity of SLAM increased with the emergence of indoor applications of robotic devices.

2.4.1 Direction of Research

For a PWU to navigate a wheelchair effectively through public spaces, they need to be able to avoid colliding with people or obstacles. An accurate map of the surroundings is key to solving this issue. However, some techniques rely on pre-existing maps of the area. A PWU may navigate their wheelchair to new locations, and can not rely on pre-existing maps for accurate navigation. Rather, the goal is to build up a real-time map of the surroundings that is accurate enough to avoid collisions.

2.4.2 Review of Existing Methodologies

A review of SLAM techniques can be found in [22], which also outlines the standard formulation of the SLAM problem as that of a Maximum a posteriori (MAP) estimation. The formulation relies on Bayes theorem, and using the prior knowledge of the robots pose to maximize the likelihood to estimate the current position of the robot. The variables required to estimate the position are the robot poses, the position of landmarks and the calibration parameters of the sensors.

In order to build an accurate map of the surroundings, the calibration of the sensors providing the measurements is a crucial step. The choice of sensors also matter, as the type of data returned by the sensor may affect the computational complexity of the SLAM algorithm. As such, it is common to have a module in the system that deals with the extraction of relevant features from the sensor data.

A fairly common assumption in SLAM approaches is that the world is static and remains unchanged as the robot moves. This becomes an issue with the goal of this project, which hopes to achieve the ability to detect human objects walking around the wheelchair. This issue will be addressed in a later section.

Visual SLAM

Visual SLAM (vSLAM) is an implementation of SLAM that relies on visual inputs only. As stated in [23], vSLAM is suitable for AR due to the low computational algorithms that can be implemented on the limited resources of an AR headset. The technique of vSlam is mainly composed of three modules:

Initialization In the initialization stage, camera pose estimation is conducted, to transform objects in a 2D image from the camera into a 3D co-ordinate system that the robot understands. This process determines the position and orientation of the camera relative to the object. A part of the environment is reconstructed as part of the initial map using the global co-ordinate system of the robot.

Tracking Here, the reconstructed map is used to estimate the pose of the camera with respect to the map. Feature mapping or feature tracking is conducted on the images in order to get a 2D-3D correspondence between the image and the map. The camera pose can then be calculated from the correspondences by solving the Perspective-n-Point problem [24]. This allows the system to identify where on the map the robot currently is.

Mapping When the robot passes through an environment that has previously not been mapped, the 3D structure of the surroundings is calculated from the camera images. The structures are then added to the existing map of the environment.

2.4.3 Comments

Due to the freedom in movement of an augmented reality headset camera, a system that relies solely on visual inputs may not be able to detect all obstacles in the surroundings. For instance, a limitation is that the PWU will not be able to extend their head backwards to view objects behind them. As such, it becomes important to consider the sensors available on powered wheelchairs, and utilize them to build an accurate map of the surroundings.

2.5 Augmented Reality Headsets

The improvements in augmented reality technology has spurred research into the use of AR devices in everyday tasks. The availability of commercial devices has also encouraged developments in the field, with products such as the Microsoft Hololens and the Magic Leap One.

2.5.1 Direction of Research

The augmented reality system built for this project needs to be able to give visual prompts to the PWU. As such, a device that already has the ability to create holograms is key. Furthermore, most AR devices have built in cameras to perceive the world around the user. We hope to be able to access the cameras on the device to do object detection and tracking.

2.5.2 Review of Existing Methods

Microsoft Hololens

The Microsoft Hololens is an untethered holographic computer, allowing for the display of 3D holograms pinned to real world objects. The Hololens is equipped with an array of sensors, making it an ideal choice of hardware for this project.

Holograms The Microsoft Hololens is able to blend real world and virtual content into environments where digital and physical objects can co-exist and interact. The term 'Mixed Reality' was first introduced by [25], and refers to the blending of the physical and virtual worlds.

The Hololens allows the developer to create 'Holograms', which are objects of light and sound that are displayed by the headset. Users are able to interact with the holograms through voice, gaze and gestures. Enhanced environment apps are applications that facilitate the placement of digital information on the user's current environment [26]. An example of an enhanced environment application is placing markers in augmented reality on objects that the user can interact with in both the physical and digital worlds.

Hardware Specifications As part of our research, we highlight the sensors on the device that may be relevant to the project. A full hardware specification is available online [27].

- 1 Intertial Measurement Unit (IMU)
- 4 Environment understanding cameras
- 1 Depth Camera
- 1 2MP Photo/HD video camera

Most importantly, the Hololens has a video camera. Preliminary research shows that it is possible to access the camera data directly, making it a suitable choice for the project.

Personal Robotics Lab The use of augmented reality devices to help PWUs is a research topic actively pursued by members of the Personal Robotics Lab at Imperial College London. Previous work has explored the use of augmented reality as a visualization tool to help PWUs understand the system dynamics of the wheelchair they operate, displaying visual cues that indicate the direction of travel of assistive control [28]. Other work involves using the camera to detect objects of interest in the environment, and developing a system that navigates the wheelchair to the detected objects through gaze and eye tracking [29].

Figure 2.5: AR visualizations and markers for interaction [29]

2.5.3 Comments

Although other AR-devices exist on the market, due to the availability of the Microsoft Hololens in the Personal Robotics Lab, as well as the research done by individuals, it is in the best interests of this project to use the Hololens as the main augmented reality device for this project.

Chapter 3

Requirements Capture

3.1 Project Deliverable

The objective of this project is to develop an augmented reality system that can be used by powered wheelchair users (PWUs) to assist them in navigating their powered wheelchairs in public spaces with many people walking in the surroundings. The system should be able to detect the presence of individuals and infer their position relative to the PWU, and by extenions, estimate their direction of travel.

We propose a system that uses the Microsoft Hololens augmented reality headset as the main input and visualization tool. The PWU would wear the headset as they operate the powered wheelchair, allowing the system to create visualizations of potential obstacles and collisions. Furthermore, the system would also encompasses the reactive control aspect of the powered wheelchair. Should an individual be detected as walking in the wheelchairs trajectory, the system will send control signals to the powered wheelchair to slow down or completely stop depending on how far the target is from the wheelchair.

By definition of the requirements, we can divide the project into three parts: Human Detection and Direction, Obstacle Mapping & Visualization, and finally Reactive Control.

3.2 Human Detection and Direction

The requirements of the Human Detection and Direction (HDD) system is to be able to use a video stream of the surroundings to determine the position and direction of people. The Hololens has a built in camera that can be used to take photos of the surroundings of the user [29]. We will leverage this ability to create a video stream.

The actual HDD sytem is implemented on another computer with access to a GPU. We utilize the GPU to be able to do real-time object detection and pose estimation of detected individuals. The system should be able to infer the direction individuals are walking in, as well as their real-world positions relative to the PWU.

Features

- Creating a live video stream using camera.
- Streaming the live video to accompanying computer.
- Object detector trained on humans/pedestrians.
- Object tracker to track detected humans, and determine their direction.
- Body/Head pose estimator to determine direction of travel.
- Stream detections back to the Hololens for visualization.

3.3 Obstacle Mapping & Visualization

This project utilizes the Microsoft Hololens as a visualization and spatial mapping tool. The HDD system will output its detections and directions to the Hololens, which is used to create visualizations that will help the PWU. Examples of the visualzation include arrows that indicate direction of movement, as well as alerting the user to potential collisions.

Features

- Receving detection/direction data from HDD system.
- Utilize Camera to World transforms of the Hololens Camera to get World coordinates of people.
- Create holographic visualizations to help PWU understand the direction people are walking in.
- Create a map of obstacles for Reactive Control.

3.4 Reactive Control

The powered wheelchair (ARTA) available in the Personal Robotics Lab (PRL) can be manually operated using a joystick. The goal of the project is for the PWU to be able to wear the Hololens as an aid for navigation in public spaces. As such, it would be beneficial for the PWU if the wheelchair had the ability to reactively control the device to prevent collisions with detected objects.

Features

- Receving object detections in front of wheelchair.
- Prevent wheelchairs from driving into objects.

Chapter 4

Analysis and Design

This chapter gives an overview of the overall system and explains the design choices made. Throughout the project, we explored various methods to implement a real-time augmented reality system for PWUs operating a wheelchair. Naturally, the structure and goals of the project have developed since the interim report, and we review the differences between the initial goals and final product .

4.1 Design Overview

As stated in the requirements, this project consists of three major components:

- Human Detection and Direction (HDD)
- Object Mapping and Visualization (OMV)
- Reactive Control on ARTA

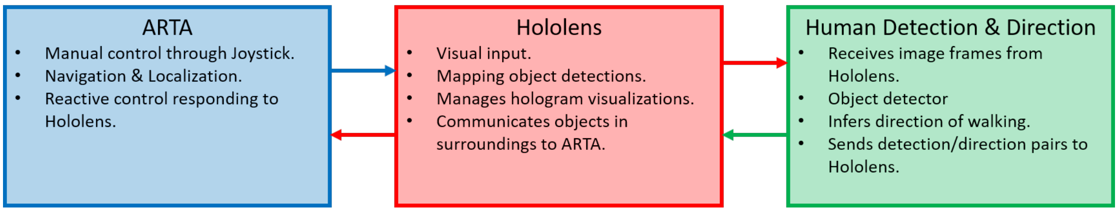


Figure 4.1: Simplified high level system diagram

From a very high level view, we can map these requirements to the respective devices they will be operating on. The HDD system takes implements the object detection and human direction inference, while the Hololens is responsible for utilizing the spatial mapping to obtain world positions of the detections, as well as visualizing the detections. The powered wheelchair (ARTA), has manual input that is overridden by the reactive control system that is dependent upon the detections and mappings. The diagram in Figure. 4.1 shows an overview of the system, and shows that the Hololens acts as an intermediary between ARTA and the HDD.

4.1.1 Hardware

	ARTA	**Hololens**	**HDD**
Hardware	Powered Wheelchair controlled by Laptop	Hololens	Desktop PC with GPU
Operating System	Ubuntu 16.04	UWP	Ubuntu 16.04

Table 4.1: Hardware description of system

Table. 4.1 summarises the hardware overall system is implemented on. The powered wheelchair, ARTA, is controlled by a laptop, which is responsible for the wheelchair speed, wheel rotations, navigation and localisation. The Hololens is a self contained augmented reality headset, running the Universal Windows Platform (UWP) operating system. Finally, the Human Detection & Direction system is implemented on a desktop computer with a GTX 1050Ti GPU, allowing it to run real time object detectors.

4.1.2 System Communication

Robotic Operating System

Since the project spans multiple operating systems, we have chosen to utilize the Robotic Operating System (ROS) as a means of communication between the devices. In ROS, a *node* is defined as a process that performs a computation. A node can be made up of smaller nodes that perform specific computations that serve the needs of the parent node.

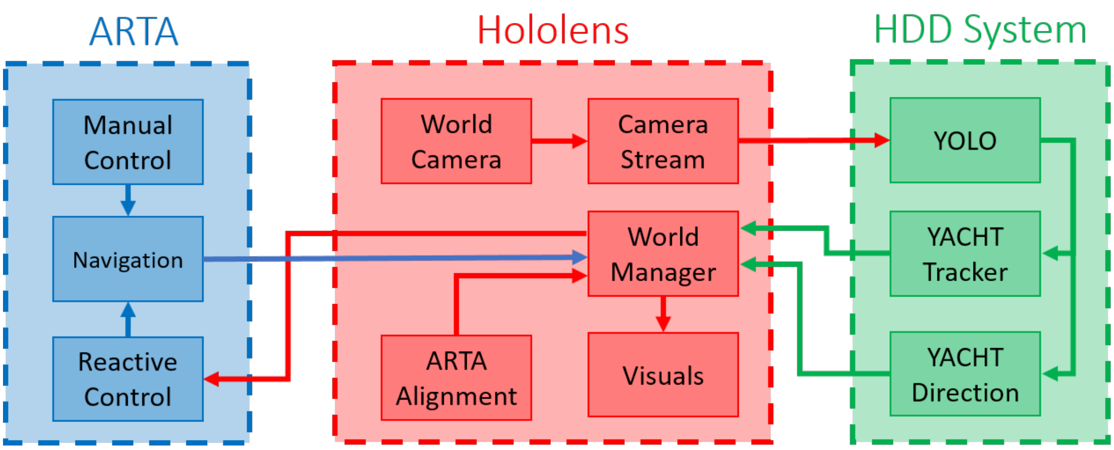


Figure 4.2: System diagram detailing individual components

We can think of the three major systems as large ROS nodes that consists of smaller nodes that run individiual tasks, such as creating the camera stream, or detecting objects. We visualize the breakdown of the system into nodes in Figure. 4.2.

ROS Topics Nodes in ROS communicate with one another by publishing data in the form of *messages* which get broadcasted over a *topic*. Nodes can choose what data they receive by subscribing to topics. This method allows for nodes running on different devices to communicate with each other, regardless of the operating system. The nodes are unaware that the data it receives is published from a node running on a seperate computer, making ROS a perfect choice for communication in this design.

4.2 Human Detection & Direction System

The HDD system is reponsible for detecting and predicting the directions of people in the surroundings of the wheelchair. By taking visual inputs in the form of images from the Hololens, we run an object detector trained on a dataset of pedestrians to detect people and heads. The bounding boxes produced by the object detector are fed as inputs to an object tracker and a body pose estimator. We use the results of these two nodes to infer the direction a detected person is moving in, and publish the results back to the Hololens.

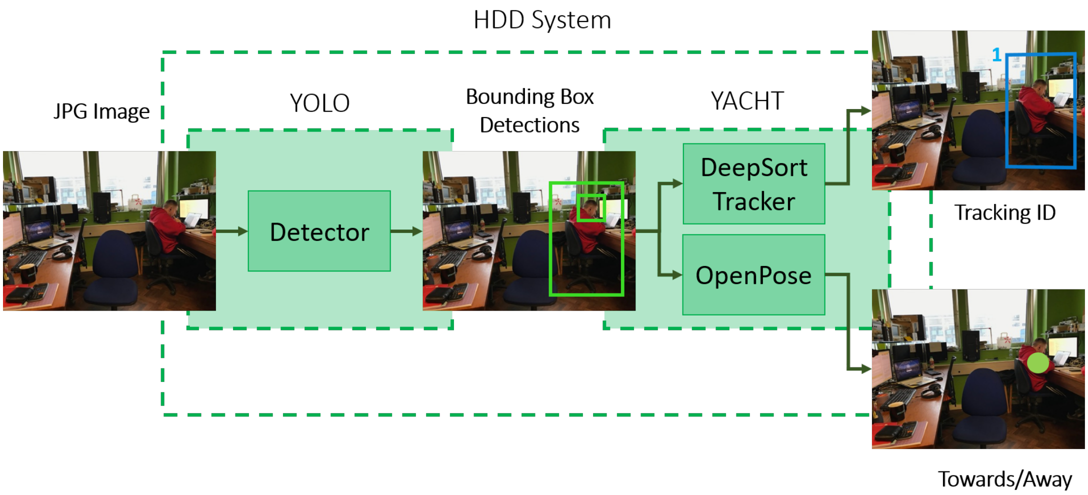


Figure 4.3: Individual nodes in HDD System

We present an overall view of the HDD System, covering the purpose and design of individual components. We also propose the reasoning behind certain design choices, which we cover in more depth later in this report.

4.2.1 YOLO Object Detector

Object detectors often form the input to an object tracker or pose estimation system [10, 30]. In the case of top-down body pose estimation methods, detections can be the first point of failure [31]. As such, the accuracy of the chosen object detector must be considered, together with the choice of using a pre-trained model or training on a more relevant dataset. Finally, we must also consider the use-case of the detector, which must be able to operate in real-time and detect moving objects as they pass by.

Choice of Detector

As commented on in Section 2.1.3, modern deep learning techniques outperform classical object detectors in accuracy, but are limited by the requirement of a GPU to perform in real-time. Since the Hololens does not have built in support to run object detection networks, Microsoft provides the Azure Cognitive Services API to allow developers to query their system for object detections. The limitation is that this service is not free, and abstracts away the implementation of an object detector. Furthermore, one of the personal goals for this project was to learn more about CNNs in computer vision.

Taking this into account, we compared several deep learning architectures for object detection. Previous work done in the PRL used Facebook AI Research's (FAIR) Detectron to detect objects [29, 32, 33]. Further discussions with members of the Imperial Computer Vision & Learning Lab suggested the use of the YOLO object detector [3], due to its speed and having a lightweight implementation that could be run on lower end GPUS at relatively high frame rates. This prompted the design decision to use the Darknet framework to run the **YOLOv3-tiny** architecture as the object detection method of choice for this project [34].

Pre-trained Model vs Training

An advantage of using the YOLO Darknet framework is that it provides models which are already trained to detect multiple object classes, including the class *Person*. One of the pre-trained models is the YOLOv3-tiny architecture trained on the Common Objects in Context (COCO) dataset [35].

Comparing Models To compare the accuracy of the bounding boxes produced by pre-trained model, sample videos were recorded using the Hololens and used as a base comparison point. It was quickly shown that although the COCO trained model can detect individuals, or multiple people who are well spaced out, it had difficulty in differentiating between people who are close together or slighlty occluded. Figure 4.4 highlights the issue of the COCO model failing to detect small people close together.

(a) COCO YOLOv3-tiny (b) Trained YOLOv3-tiny

Figure 4.4: Model comparison on a video from Sherfield Walkway at Imperial College London

Pedestrian Dataset A common case in pedestrian detection is occlusion, where only certain body parts are visible. To resolve this issue, we decided to train the YOLOv3-tiny model on the **CrowdHuman** dataset [36], which contains annotated images of crowds, including bounding boxes for the head, visible human region and full-body. The annotations for partially visible people allows the network to detect occlusions, solving the issue of people being close to each other. We also trained the network to detect heads, since we initially wanted to use head pose estimation to determine direction.

Analysis The result of training the YOLOv3-tiny model on the CrowdHuman dataset is that the system is able to detect smaller figures who are occluded. The additional ability to detect heads allowed us to explore the use of head pose estimation for direction inference. We go into further detail on the training process in the Implementation section of this report.

4.2.2 YACHT: Yet Another Crowd Human Tracker

The bounding boxes produced by the object detector are consumed by the **Yet Another Crowd Human Tracker** (YACHT) module. As shown in Figure 4.3, we can see that the YACHT module is made up of two seperate nodes. The tracker node uses the Deep SORT algorithm to track detected individuals [12], while the pose estimator node uses the OpenPose [20] network to determine whether a person is walking towards or away from the PWU.

In the following sections, we briefly explain the methods used to infer the directions people are walking in. We also explore the use of head pose estimation, and the limitations that prevented it from making it to the final product.

YACHT Tracker: Object Tracking

We explored existing object tracking methods in Section 2.2.2. We further discussed our choices in 2.2.3, where we express the need for an online object tracking system. As such, we chose to investigate two related methods, SORT [10] and Deep SORT [12].

SORT The Simple Online and Realtime Tracking (SORT) method is a fast online object tracker. The initial implementation of YACHT used the SORT algorithm due to its speed. However, it was quickly realized that due to the simplicity of the association metric, object tracking was not very accurate, especially for occluded objects. When two objects crossed paths, the tracker was unable to recognize the act, and re-labeled the objects with brand new tracking IDs.

Deep SORT Deep SORT is an extension of the original SORT algorithm, but uses a deep network to generate feature descriptors for the predicted bounding boxes. We explain the algorithm in Section 2.2.2, but to repeat, instead of using the Intersection-over-Union association metric to compare bounding boxes, a Neareast-Neighbours is used to compare the feature vector of the new detection with a library of feature vectors for the tracked object. This additional step allows for the problem of occlusions to be reduced.

Figure 4.5: MOT16 benchmark [37] (L) using our YOLO model (M) for Deep SORT (R)

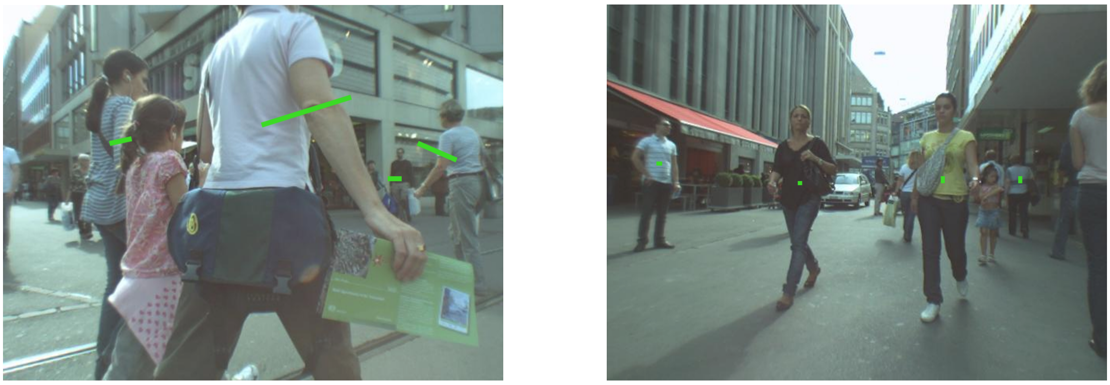
Analysis The generation and storage of feature descriptors for tracks is an expensive process. For the system to run in real-time, a GPU is needed to accelerate the network. We have made changes to the Deep SORT implementation so it can run on Tensorflow-GPU, which we explain later in this report. This improves the speed significantly, but uses up precious memory. As a result, we had to consider the amount of memory avaiable on the GPU, since the YOLO detector and OpenPose networks also rely on GPU acceleration. After testing, we found that it was possible to run both networks on the GPU at the same time, and we chose to use the Deep Sort method.

Object Tracking for Direction Inference

An idea we explored was to use the previous image co-ordinates of each detection to predict the direction a person would walk in. This involved storing the previous states of each track and extrapolating the centroids of each track to determine a direction.

Algorithm For each tracked object in a frame:

Data: (x,y) centroid image co-ordinates up to the previous 5 states
Result: (x,y) of extrapolated point
for *Frame* **do**
  for *Tracker* **do**
    if *Tracker existed in previous frame* **then**
      Extrapolate over the centroids of previous states;
      Return linear extrapolation (x,y);
    end
    if *Tracker has no previous states* **then**
      Add (x,y) centroid of tracker to queue of previous states;
      Return current centroid (x,y);
    end
  end
end



(a) Tracks with horizontal motion (b) Tracks with mostly vertical motion

Figure 4.6: Linear extrapolation works for objects that move across the frame, but it becomes difficult to determine the direction when mostly vertical motion occurs

Issues The problems with this method occurs when the centroid of the tracked objects does not move much horizontally. As such, linear extrapolation is not possible, and this leads to an ambigious definition of the direction. Since the object is not moving across the screen, it is not possible to differentiate between a person standing still, moving directly towards the PWU or walking away.

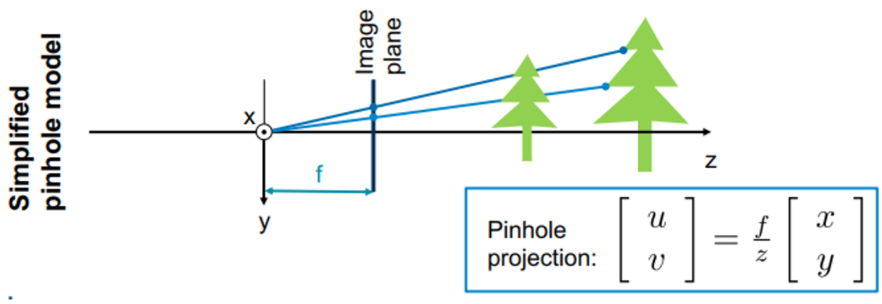


Figure 4.7: Pinhole Camera projection and the loss of the z-axis [38]

In the pinhole camera model, when an object in the real world is projected onto a 2D image, we lose the distance along the z-axis. The issue with world-to-camera projection is when we want to obtain the world co-ordinate of the object, the best we can do is to calculate a ray through the 2D image point. However, as shown in Figure 4.7, it becomes impossible to tell how far away the object is without a depth camera, since the object could exist anywhere along that ray.

YACHT Direction: Body Pose Estimation

To solve the direction ambiguity brought up in Section 4.2.2, we proposed the use of body pose estimation techniques to determine whether a person is walking towards or away from the camera of the PWU. We researched several body pose implementations in Section 2.3.2, but we ultimately decided on OpenPose, due to its well documented implementation on Github [20].

Object Detectors & Bottom-Up Approaches The output of the YOLO object detector is the original image and the associated bounding box co-ordinates of detections. Since OpenPose is a bottom-up approach to the body pose estimation problem, we admit that it is counter-intuitive to use an object detector to detect individual people when OpenPose determines the body part keypoints across the whole image. This will be explored more in the evaluation of the report.

Keypoint Estimation We use OpenPose to estimate the positions of the keypoints. The Github implementation provides several pre-trained models for body pose estimation. From our tests, we found that the *BODY_25* model was the fastest, suiting our real-time requirements. Figure 4.8 shows the keypoints generated by the model.

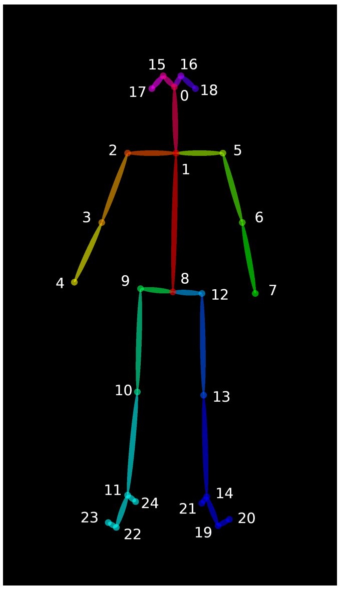


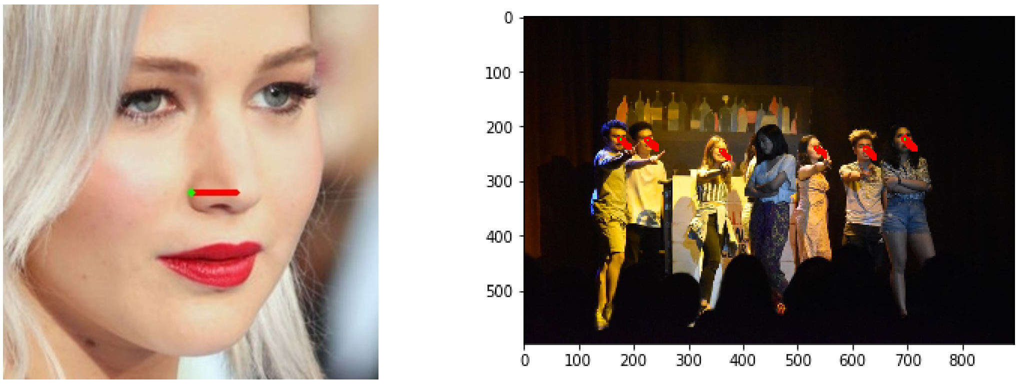
Figure 4.8: Keypoints produced by the BODY_25 model [20]

The model is able to differentiate between the left and right limbs of the human body, making it a suitable choice to determine if a person is facing the camera or not. We further explain the methodology in the implementation section of the report.

Head Pose Estimation

We initially began the project by exploring the use of head-gaze estimation as another way of infering where someone will walk. We researched the concept of head pose estimation in Section 2.3.2, with the logic being people tend to look in the direction where they are walking. We leveraged the use of the DeepGaze library as an initial starting point [39], since the library has a built in head-pose estimator.

Reasons for removal We noticed from our research that the head pose estimation was not very accurate, as can be seen in Figure 4.9. A close up image of a face still returns an inaccurate estimation of the head pose. For smaller faces with multiple detections, the head pose estimation was not performed in real-time and once again had inaccurate



(a) Head pose estimation on close-up of face (b) Head pose estimation on smaller people

Figure 4.9: DeepGaze head pose estimator

estimations. Finally, as we will explain in the implementation section, the quality of the images received from the Hololens was too low, and as such, facial landmark detectors required for head pose estimation were unable to detect the keypoints.

4.3 Hololens

4.3.1 Breakdown

4.4 ARTA

4.4.1 Breakdown

Bibliography

[1] Ya Li Hou and Grantham K.H. Pang. Human detection in crowded scenes. *Proceedings - International Conference on Image Processing, ICIP*, (September 2010):721–724, 2010.

[2] Dongdong Zeng, Ming Zhu, Tongxue Zhou, Fang Xu, and Hang Yang. Robust Background Subtraction via the Local Similarity Statistical Descriptor. *Applied Sciences*, 7(10):989, 2017.

[3] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. Technical report.

[4] Massimo Piccardi. Background subtraction techniques: a review*. 2004.

[5] Manato Hirabayashi, Shinpei Kato, Masato Edahiro, Kazuya Takeda, Taiki Kawano, and Seiichi Mita. GPU Implementations of Object Detection using HOG Features and Deformable Models. Technical report.

[6] Paul Viola and Michael Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. Technical report, 2001.

[7] Navneet Dalal, Bill Triggs, Navneet Dalal, and Bill Triggs. Histograms of Oriented Gradients for Human Detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[8] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[9] Caglayan Dicle, Octavia I Camps, and Mario Sznaier. The way they move: Tracking multiple targets with similar appearance. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2304–2311, 2013.

[10] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *Proceedings - International Conference on Image Processing, ICIP*, volume 2016-Augus, pages 3464–3468, 2016.

[11] R. E. Kalman and R. S. Bucy. New Results in Linear Filtering and Prediction Theory. *Journal of Basic Engineering*, 83(1):95, 1961.

[12] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *Proceedings - International Conference on Image Processing, ICIP*, volume 2017-Septe, pages 3645–3649, 2018.

[13] Roberto Valenti, Nicu Sebe, and Theo Gevers. Combining head pose and eye location information for gaze estimation. *IEEE Transactions on Image Processing*, 21(2):802–815, 2012.

[14] Erik Murphy-Chutorian and Mohan Manubhai Trivedi. Head pose estimation in computer vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):607–626, 2009.

[15] Vahid Kazemi and Josephine Sullivan. One Millisecond Face Alignment with an Ensemble of Regression Trees. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1867–1874, 2014.

[16] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-Janua, pages 3711–3719, jan 2017.

[17] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 770–778, 2016.

[19] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter Gehler, and Bernt Schiele. DeepCut: Joint subset partition and labeling for multi person pose estimation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 4929–4937, 2016.

[20] Zhe Cao, Tomas Simon, Shih En Wei, and Yaser Sheikh. Realtime multi-person 2D pose estimation using part affinity fields. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-Janua, pages 1302–1310, 2017.

[21] T Bailey and H Durrant-Whyte. Simultaneous localisation and mapping (SLAM): Part I - The essential algorithms. *IEEE Robotics and Automation Magazine*, 13(2):99–108, 2006.

[22] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, Jose Neira, Ian Reid, and John J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.

[23] Takafumi Taketomi, Hideaki Uchiyama, and Sei Ikeda. Visual SLAM algorithms: a survey from 2010 to 2016. *IPSJ Transactions on Computer Vision and Applications*, 9(1):16, 2017.

[24] David Nistér. A Minimal Solution to the Generalised 3-Point Pose Problem. *Journal of Mathematical Imaging and Vision*, 27(1):560–567, 2004.

[25] Paul Milgram and Fumio Kishino. A TAXONOMY OF MIXED REALITY VISUAL DISPLAYS. *IEICE Transactions on Information Systems*, E77(12):1–15, 1994.

[26] Microsoft. Windows Mixed Reality Documentation, 2018.

[27] Microsoft. Microsoft HoloLens HoloLens Device Specifications. 2015.

[28] Mark Zolotas, Joshua Elsdon, and Yiannis Demiris. Head-Mounted Augmented Reality for Explainable Robotic Wheelchair Assistance. 2018.

[29] Rodrigo Chacón-Quesada and Yiannis Demiris. Augmented Reality Control of Smart Wheelchair Using Eye-Gaze-Enabled Selection of Affordances. pages 1–4, 2018.

[30] Sheng Jin, Xujie Ma, Zhipeng Han, Yue Wu, Wei Yang, Wentao Liu, Chen Qian, and Wanli Ouyang. Towards Multi-Person Pose Tracking : Bottom-up and Top-down Methods. *Proceedings of the IEEE International Conference on Computer Vision*, (2):4–7, 2017.

[31] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9910 LNCS, pages 34–50, 2016.

[32] Ross Girshick, Ilija Radosavovic, Georgia Gkioxari, Piotr Dollár, and Kaiming He. Detectron. \url{https://github.com/facebookresearch/detectron}, 2018.

[33] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2017.

[34] Joseph Redmon and Ali Farhadi. YOLOv3: An Incremental Improvement. Technical report, 2018.

[35] Tsung Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8693 LNCS, pages 740–755, 2014.

[36] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. CrowdHuman: A Benchmark for Detecting Human in a Crowd. 2018.

[37] Anton Milan, Laura Leal-Taixe, Ian Reid, Stefan Roth, and Konrad Schindler. MOT16: A Benchmark for Multi-Object Tracking. 2016.

[38] Stefan Leutenegger. Representations and Sensors. 2019.

[39] Massimiliano Patacchiola and Angelo Cangelosi. Head pose estimation in the wild using Convolutional Neural Networks and adaptive gradient methods. *Pattern Recognition*, 71:132–143, 2017.