ENERGY EFFICIENT OBJECT DETECTION AND MEASUREMENT FOR SMART GLASSES

By

Jing Yang

A THESIS

Submitted
to Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science - Master of Science

2014

ABSTRACT

ENERGY EFFICIENT OBJECT DETECTION AND MEASUREMENT FOR SMART GLASSES

By

Jing Yang

We design and implement a novel object detection and measurement system called Lockon for smart glasses. Lockon takes the advantages of the mounting position of the smart glasses, and provides users two useful functions that can benefit a wide range of applications. Lockon can accurately locate the object of interest (OoI) in the view of the user and inform the user the position of the OoI in real-time, using the front-facing camera equipped on the smart glasses and advanced computer vision and image processing techniques. To conserve energy, Lockon implements a motion trigger to intelligently activate the object detection process only when it is necessary. Lockon can also accurately measure the dimension of the object, with a 3D ranging technique. This capability allows user to remotely estimate the dimension of the object. We implement Lockon on Google Glass Explorer Edition, and evaluate the performance of Lockon using extensive experiments. Our results indicate that Lockon can achieve high detection accuracy (0.95 true positive rate and 4.3×10^{-7} false positive rate), low object dimension measurement error (3.3% when distance is less than 1 m), and low delay (300 ms).

This thesis is dedicated to someone.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisor Prof. Guoliang Xing for the continuous support of my master study and research, for his patience, motivation, enthusiasm and immense knowledge. His guidance helped me in all the time of study, research and writing of this thesis.

Furthermore I would like to thank the rest of my thesis committee: Prof. Richard J. Enbody and Prof. Xiaoming Liu, for their encouragement, insightful comments, and valuable discussions.

Finally, and most importantly, I would like to thank my husband Ruogu Zhou. His support, encouragement, patience and unwavering love were undeniably the bedrock upon which my life have been built. I would like to express my heart-felt gratitude to my parents, Jianhua Yang and Yuying Kang, for their faith in me and allowing me to be as ambitious as I wanted. Also, I thank Ruogu's parents, Hongbin Zhou and Xiaojuan Jin, for providing me with unending encouragement and support. I thank my dear daughter Eva Zhou. She spent countless hours entertaining herself while I sat at the computer typing away. You are the best little girl. I love you.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Recent years have witnessed the emergence of a new class of wearable devices, including smart watch, smart glasses, and smart bracelets for health tracking. A representative example of smart glasses is Google Glass [3]. It is estimated that the sale of smart glasses would grow from 8,7000 in 2013 to at least 10 M by 2018 [2]. Almost all the major players in the mobile industry including Google, Samsung, and Apple, have already released or reported to plan to release smart glasses products in the near future. Intergrating ubiquitous connectivity (e.g., cellular, WiFi, Bluetooth, and etc.), rich computing capability, and versatile sensing ability (e.g., camera, microphone, accelerometer, and etc.) into a pair of glasses that could be worn comfortably by users, smart glasses offer unique advantages over traditional smart devices like smartphones and tablets. The high mounting position (on user's head) provides smart glasses unobstructed view, allowing them to see what the users are looking at. As the screen of the smart glasses is mounted directly above or over the eyes of the user, it can provide the user real-time information without requiring any cumbersome between the user and the device. These traits of smart glasses open up a wide range of new applications, ranging from augmented reality, activity tracking, to vision-based crowd sensing.

A fundamental functionality that is required by these exciting new applications is real-time object detection and measurement. Sharing the view of the user, smart glasses can help user identify objects of interest in the view, and offer user helpful information accordingly. By measuring the distance between the user and the object, as well as the dimension of the object, it can enable many exciting applications in areas such as medical care, tourist guiding, and education. However, existing object detection systems [18] [14] designed for general smart devices usually incur high computational and energy overhead, thus cannot work well on smart glasses that are equipped with slower CPUs and significantly smaller batteries. Moreover, existing object detection systems are usually optimized for general-purpose smartphones and tablets, thus are cumbersome to operate

on smart glasses that lack friendly user interface. Existing remote object dimension measurement approaches [20] [21] also cannot be readily applied to smart glasses, since their operation requires either special hardware that is not available on smart glasses, or stationary fixtures that are ill-suited for mobile applications.

We propose a new real-time object detection and measurement system called Lockon for smart glasses. Lockon adopts the Voila-Jones object detection framework to achieve high computational efficiency, and the HOG feature descriptor to offer robustness against illumination variations. We extensively optimize the detector to improve its robustness to object orientation variations while minimizing the detection delay. To reduce the energy consumption incurred by object detection, we design a motion trigger to intelligently activate the object detection process if object detection is needed, and deactivate the process otherwise. Lockon accurately measures the dimension of the object using the stereo triangulation technique. We design a scheme called head-tilting for users to conveniently measure the object in close distance ($<$ 1 m).

In Summary, we make the following major contributions in this work.

1. We design as robust, low complexity, and energy-efficient object detection system for smart glasses. This system can detect various type of objects, with high accuracy and low delay. We extensively optimize the detector for improving robustness. To reduce system energy consumption, we also design a motion trigger that intelligently determines if the object detection is needed, and activates/deactivates the object detection process accordingly.

2. We design a novel object dimension measurement system, which can measure the object accurately in close range ($<$ 1 m) using 3D ranging techniques. We also design a scheme for smart glasses users to rapidly and conveniently measure the object by only tilting their heads.

3. We implement a prototype of Lockon on Google Glass Explorer Editon, and evaluate its performance using extensive experiments. The results indicate that Lockon can achieve high

detection accuracy and object dimension measurement accuracy while only incurring low detection delay.

1.1 Related Work

Real-time object detection has drawn extensive research interests for decades. D.M.Gavrila et al. proposed a method using Discrete Transformation based matching techniques to detect stop signs and pedestrians using a camera mounted on vehicle [14]. A security surveillance system [18] proposed by A. Roy et al. detects moving objects in real-time using background modeling techniques. Although these approaches show adequate performance in detecting certain objects under typical settings, they may perform poorly in detecting objects subject to large illumination and orientation variations. Moreover, these approaches require powerful computing hardware to achieve high accuracy and low delay, thus are not applicable to resource-constrained mobile systems. Various modern digital cameras [6] can accurately detect faces present in the view finder with low delay. However, the detection techniques used on cameras have been extensively optimized for detecting only faces, thus may not provide adequate performance in detecting other types of objects. Several mobile apps like LookTel [5], have been developed for general smart devices like smartphones and tablets, to detect objects of user's interest. Unfortunately, as these apps are not designed for smart glasses that have slower CPU and less energy resource, they tend to perform poorly on smart glasses. Moreover, these apps require extensively manual interaction with the device, which are ill-suited for smart glasses that are usually cumbersome to physically interact.

Remote dimension measurement has been well studied in applications like passive remote sensing [8], in which a satellite or an aircraft flies over an area and uses onboard cameras to take photos of the area. The terrain of the area can be reconstructed using the photos and advance image processing techniques. Although this technique could be applied to measure the dimension of objects, it incurs significant computational overhead in finding the matching points of the objects on photos, thus cannot be use on smart glasses. Techniques [21] commonly used in 3D object modeling could accurately measure the dimension of the modeled objects. However, most of them involve

using special hardware to generate laser beam that slowly sweep through the surface of the entire object. As a result, these techniques cannot be applied to dimension measurement on Lockon. T. Wang et al. propose an alternative approach [20] to measure the 3D dimension of a remote object with only a single camera mounted on an adjustable tripod. The stereo triangulation technique is used to measure the object dimension. However, this technique requires using a stationary tripod, thus cannot be employed by smart glasses that are highly mobile.

CHAPTER 2

BACKGROUND

2.1 Smart Devices and Wearable Devices

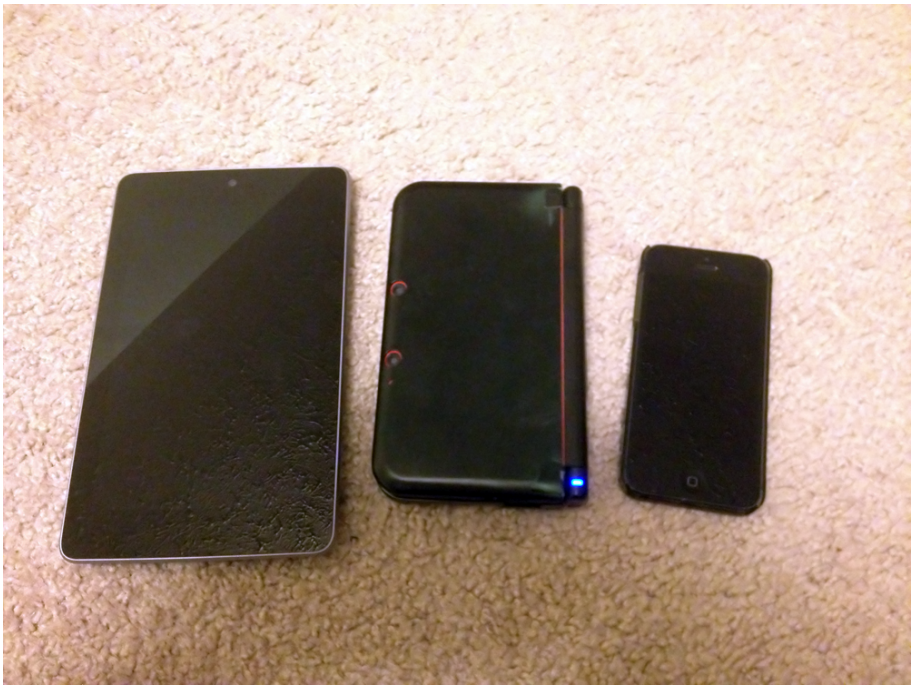


Figure 2.1: **Examples of smart devices.** From left to right: a Google Nexus 7 tablet, a Nintendo 3DS game console, and an Apple iPhone 5.

Intergrading ubiquitous connectivity, rich computing capability, and versatile sensing ability, smart devices can intelligently collect data and make decisions, enabling a wide range of novel applications. An example of such applications is *Take Runkeeper* [9], which is fitness-tracking application for Android and iOS. It tracks users' physical activities including running, walking, cycling, and hiking using various sensors including GPS and accelerometer. Smart devices are equipped with a diverse set of sensors, including camera, microphone, accelerometer, digital compass, gyroscope, GPS, and thermometer. Connectivity-wise, smart devices are commonly equipped with WiFi, Bluetooth, 3G/4G cellular interfaces. Some devices also have built-in ZigBee and NFC support. As the large number of onboard sensors and the ubiquitous connectivity could generate

large volume of data in short time, smart devices must have sufficient computational power to process the generated data. Over the last few years, the computational power of the smart devices increased rapidly, which is indicated by the fast growing CUP operating frequency and the number of CPU cores.

Figure 2.2: **Examples of wearable devices.** Including watch, wristband, and glasses. Photo is from http://bits.blogs.nytimes.com.

The advancing of smart devices triggers the emergence of a class of wearable devices [11], which can be integrated onto clothing and accessories, such as watches, glasses, and headbands, can be comfortably worn by users. Equipped with similar capabilities as general smart devices, wearable devices can perform not only tasks commonly found on general smart devices, but also specialized applications thanks to the integration of special sensors and the unique mounting positions of wearable devices. For example, in medical care applications, patient heart rate can be monitored remotely by a bracelet-like wearable device attached to the wrist of the patient. It is estimated that 90 million wearable devices will be shipped worldwide in 2014.

2.2 Smart Glasses

Smart glasses are gaining significant momentum as the market of wearable devices expends at a fast speed. It is estimated that the sale of smart glasses would grow from 8,7000 in 2013 to at least 10 M by 2018 [2]. A growing number of big players including Google, Samsung, and Apple, have entered or plan to enter this market. Between the time when the first smart glasses (Google Glass) launched and April 2014, over 20 glasses models have been launched to consumer market, with the price tags ranging from $25 to a few thousands dollars. The fast growing of smart glasses' popularity is largely due to the potential of enabling numerous new applications For example, in medical care, smart devices can be used to display real-time patient health data to doctors and nurses on-the-fly, who can then react to emergency health conditions immediately.

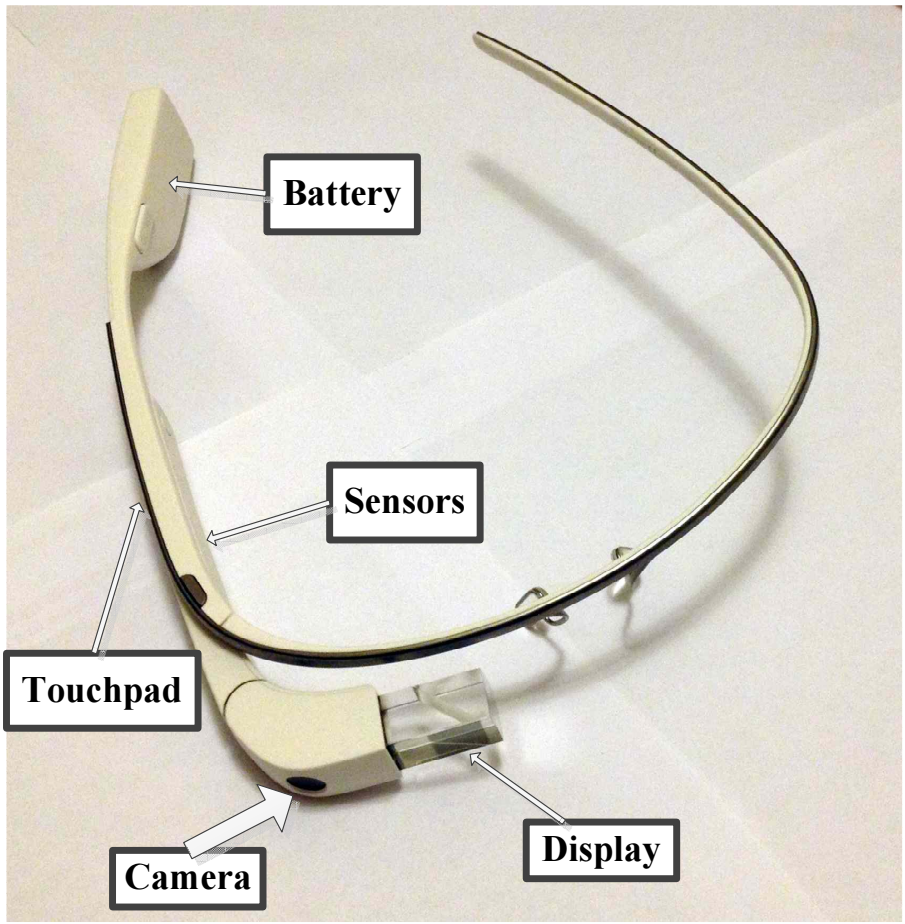


Figure 2.3: **A Google Glass Explorer Edition and its components.**

A representative example of smart glasses is Google Glass. It has a TI OMAP 4430 proces-

sor,1GB of RAM and 16GB of storage, which are powerful enough to run most applications found on general smart devices. For connectivity, Google Glass has 802.11b/g and Bluetooth radios. A small reflective surface above the eye position reflects the image projected from a small LCD inside the Glass, allowing clear display without blocking the normal vision of the user. Besides the sensors commonly available on general smart devices (accelerometer, gyroscope, compass, etc.), it also has a wink sensor that can detect eye blinking of the user, which enables a wide range of new applications like fatigue measurement and provides a new user interface. A touchpad on the side of the Glass enables easy interaction, although the Glass could be also controlled by voice command. The Glass also equips a front-facing camera besides the reflective surface. Due to the unique mounting position (on user's head), Google Glass is always facing the same way with the user's head, and is almost always horizontally positioned. This trait of Google Glass significantly benefits certain applications such as realtime object tracking/detection, reality augmentation, and activity tracking.

2.3 Object Detection

Object detection is the task of finding the objet of interest (OoI) in an image or video sequence, using object models which are known prior. Usually, object detection algorithms use extracted features and learning algorithms to recognize instances of an object category. Although it has been studied for years, a robust, accurate and fast object detection approach is still a great challenge today. Many factors can affect the detection performance, such as the amount of visual features of target object, training image quality and quantity, and the characteristics of the detection algorithms. Moreover, object may look significantly different under varying environmental factors, such as illumination, viewing perspective, and distance.

Popular object detection algorithms fall into 3 basic categories. Geometery-based approaches employ the 3D geometric models of the OoI to deal with the appearance variation caused by varying perspective and illumination. Appearance-based approaches utilize advanced feature descriptors and patterns recognition algorithms to find the shape and the texture of the OoI. The feature-

based approaches discover the interesting points of the OoI that are insensitive to scale, perspective, and illumination changes.

Object detection is fundamental to many important applications, including industrial vision-based control (e.g., robot control on assembly lines), and human-computer interface (e.g., Microsoft Kinnect). Recently, the increasing popularity of smart devices has promoted the wide adoption of object detection techniques in mobile apps development. Two examples of these apps are diet tracking, and camera-based vehicle adaptive cruise control. Due to the popularity of objective detection techniques on smart device development, many computer vision toolboxes, such as OpenCV and FastCV, now offer smart device support to facilitate the mobile app development. The object detection implemented in our system is based on OpenCV.

2.4 OpenCV: Open Source Computer Vision

Open Source Computer Vision (OpenCV) is an open-source computer vision library for real-time image processing. It offers more than 2,500 computer vision related algorithms, and supports a wide range of operating systems, such as Windows, Linux, Mac OS X, Android and iOS. It is widely adopted in real-time computer vision applications, including camera calibration, face recognition, gesture recognition and motion tracking.

CHAPTER 3

CHALLENGES AND SYSTEM OVERVIEW

3.1 Challenges

As mentioned in Section.1, the mounting position of smart glasses provides smart glasses several unique traits. Lockon takes advantage of these traits and provides users two useful functions that can benefit a wide range of applications. Lockon can accurately locate the object of interest (OoI) in the view of the user and inform the user the position of the OoI in real-time, using the front-facing camera equipped on the smart glasses and advanced computer vision and image processing techniques. Moreover, Lockon can also accurately measure the dimension of the object, with a 3D ranging technique. This capability allows user to remotely estimate the dimension of the object.

The applications of Lockon can be found in areas such as medical care, augmented reality, tourist guide, and education. For example, Lockon can be employed by automatic museum touring guide systems, which identify and locate the exhibitions appeared in the view of the tourist, and offer useful information about the exhibitions to the tourist. Lockon can be also used in wildlife observatory, in which Lockon detects and identifies the wild animal present in the user's view, and measures the size of the animal as well as the distance to the animal.

Several challenges must be addressed in the design of Lockon. First, smart glasses usually have very tight energy budget due to their small form factor. However, object detection and imaging processing are intrinsically compute intensive. Processing images taken by the camera at high rate in real-time consumes large amount of energy. Lockon must be able to minimize its energy consumption without scarifying the detection performance. Second, as the OoI can be presented to Lockon in a variety of illumination conditions and orientations in practice, Lockon must be robust to these variations. Although complex detector and extensive image pre-processing could help improve the detector's robustness to these variations, they tend to incur long detection delay. As

a result, achieving robustness in object detection while incurring low delay is challenging. Third, to accurately measure the dimension of the object, the distance between the camera and the object must be measured first. However, the sensors on typical smart glasses cannot measure distance.

3.2 System Overview

Fig. 3.1 illustrates the system architecture of Lockon, which operates as an application on the glasses OS and interacts with the hardware components, such as sensors and display, via system API calls.

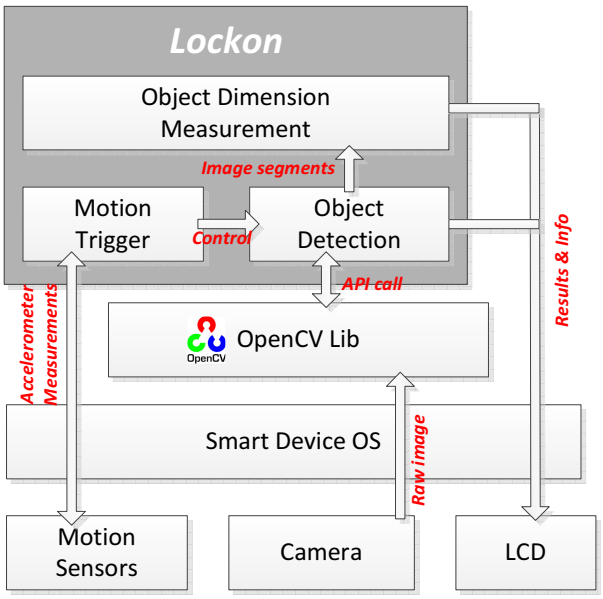


Figure 3.1: **Architecture of Lockon in on Android platform.**

As shown in Fig. 3.1, Lockon is composed of three major components, namely *objective detection*, *motion trigger* , and *object dimension measurement*. The object detection module implements the fundamental detection function of Lockon using OpenCV libraries. It employs the detectors trained offline to determine if the image frames fetched from the camera of the smart device contain the OoI. Lockon adopts the Viola-Jones detection framework (cascade detection) for improving detection performance and reducing computational overhead. To provide detection robustness to object illumination variations, Lockon employs HOG (Histogram of Oriented Gradients)[13] feature descriptor which performs localized image normalization to enhance image contrast. Ro-

bustness to object orientation change is achieved by utilizing multiple detectors that handle objects viewed from different perspectives. After detection, the image segments that contain the OoI are returned by the detector, and are marked on the screen of the smart device to indicate the locations.

Cascade detectors use a sliding window to scan over the image for OoI at multiple scales. As a result, it incurs significant computational overhead to process images captured from the camera at 30 frame-per-second. As a result, object detection process should be turned off when it is not being used by user to conserve the energy of the smart glasses. The motion trigger module of Lockon serves as a switch to intelligently activate the object detection process only when it is necessary. Specifically, Lockon utilizes the accelerometer that is commonly available on smart glasses to monitor the motion of the user, and intelligently determines when to activate the object detection process. This motion triggered detection control scheme does not require users to physically interact with the device, which greatly benefits the usually diminutive wearable devices that are cumbersome to operate.

Remotely measuring the dimension of the OoI is very useful for applications like navigation and tourist guiding. Although the size of the detected segments contain useful information about the dimension of the OoI, accurately calculating it requires knowing the distance between the detected object and the smart device. Based on the detection results, Lockon utilizes a 3D ranging technique called stereo triangulation to accurately measure the distance, as well as the dimension of the detected object. Specifically, the detection results of two frames taken at different locations are analyzed, from which the distance to the detected object and the dimension of the object are computed.

We implemented Lockon on Google Glass Explorer Edition running Android 4.0.3. The UI of Lockon is shown in Fig. 4.5. The yellow boxes on top of the UI indicates the locations of detected OoIs, with the labels indicating the type of the OoI. The green progress bar on the bottom of the UI indicates the status of motion trigger. The length of the bar is proportional to the duration that the device has been staying still since last movement. A full length of the progress bar indicates that the object detection is ongoing.

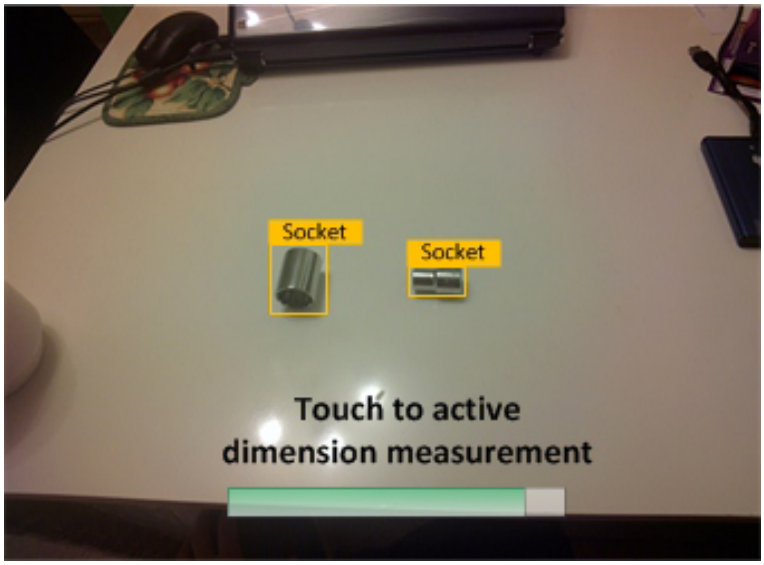


Figure 3.2: **User Interface of Lockon.** The objects detected are two wrench sockets used for mechanical work.

CHAPTER 4

ROBUST OBJECT DETECTION

The most fundamental task of this work is to design a robust object detection system for smart glasses. We have a few design objectives. First, the system should be robust, i.e., achieving satisfactory performance (low false positive rate and high true positive rate) regardless of environments (e.g., illumination) or object orientation. Second, the detection algorithm should have low computational complexity, since the object detection is performed on smart glasses which have limited computational resource and tight energy budget. Third, the detection delay should be within acceptable range ($<$1s). Achieving all of these objectives are challenging on resource-limited smart glasses, requiring carefully design and optimization of the detector.

4.1 Viola-Jones Cascade Detector

Lockon adopts the Viola-Jones object detection framework ("Viola-Jones framework")[19] due to its high accuracy and low computational overhead. Proposed by Paul Viola and Michael Jones in 2001, it is the first object detection framework that could offer real-time object detection with favorable accuracy. Due to the use of cascade classifiers, the detection process can be performed with low latency, although it tends to incur high computational overhead during training. Another advantage of Viola-Jones framework is its versatility in detecting various types of objects, although it was originally proposed to for face detection problems.

Most commonly used object detection algorithms employs window sliding through all the regions of the image that may contain the OoI. Typically, a sample is created from the window after each sliding, and processed by the detector. Due to the high computational complexity, detection algorithms proposed prior to Viola-Jones framework struggle to process video in real-time with acceptable detection accuracy. Viola-Jones framework utilizes cascade detector to accelerate the detection, which is a type of ensemble learning process. Different from other multiexpert based

ensemble detectors (e.g., voting and stacking) that are constructed with strong detectors running in parallel, cascade detector is constructed by concatenating weak detectors. The unique advantage of the cascade detector over multiexpert detector is its low computational complexity. In multiexpert ensemble algorithms, each strong detector processes all the features of a sample to detect the presence of the object, and the final decision is made by voting or stacking. As the strong detectors are generally complex, multiexpert detectors incur high computational overhead and hence are usually slow. On the other hand, cascade detectors adopt weak detectors that only examine a subset of all features at every stage, and utilize the output from the previous stage to facilitate the detection of the next stage. Specifically, the samples that are classified as negative at previous stages are discarded and excluded from the samples to be processed in following stages. To further improve efficiency, the weak detectors are arranged in ascending order in terms of complexity. As a result, negative samples that are easy to determine are quickly discarded by the weakest detectors, leaving only a few difficult samples to be processed by the stronger, more complex classifiers. This technique significantly increases the processing speed without harming the performance much.

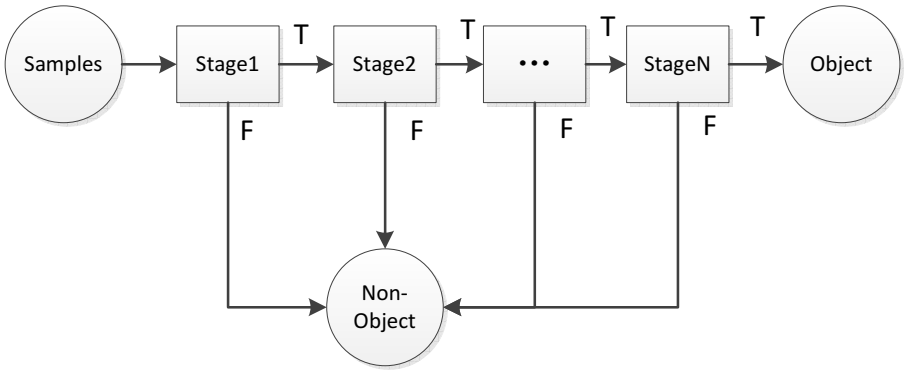


Figure 4.1: **An illustration of cascade detector.**

We now show two key properties of cascade detectors. For a cascade detector with N stages, the overall true positive rate, D, and overall false positive rate, F, can be expressed as:

$$D = \Pi_{i=1}^{N} d_i \tag{4.1}$$

$$F = \Pi_{i=1}^{N} f_i \tag{4.2}$$

where d_i and f_i are the true and false positive rates of stage i, respectively. From 4.1 and 4.2, we can observe two interesting properties of cascade detectors. For achieving a low overall false positive rate F, each stage can have a poor false positive rate f_i. For example, if per stage false positive rate is 50%, then the overall false positive rate would be around 10^{-6} when $N = 20$, which is sufficiently good for most tasks. However, for achieving acceptable overall true positive rate D, the per stage true positive rate should be sufficiently close to 1. For example, to achieve an overall true positive rate of 90% with 20 stages, each stage should have a true positive rate of at least 99.4%. To improve detection performance without incurring high computational overhead, the cost-aware ADAboost based algorithms [15] are adopted as the weak detector for every stage.

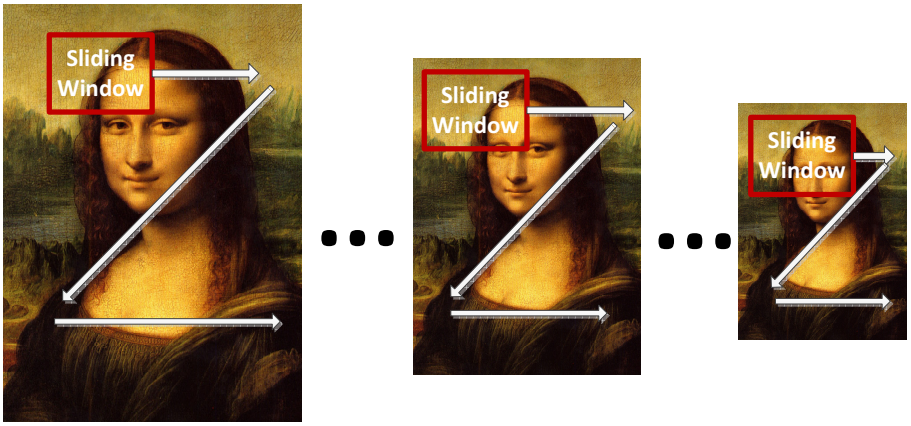


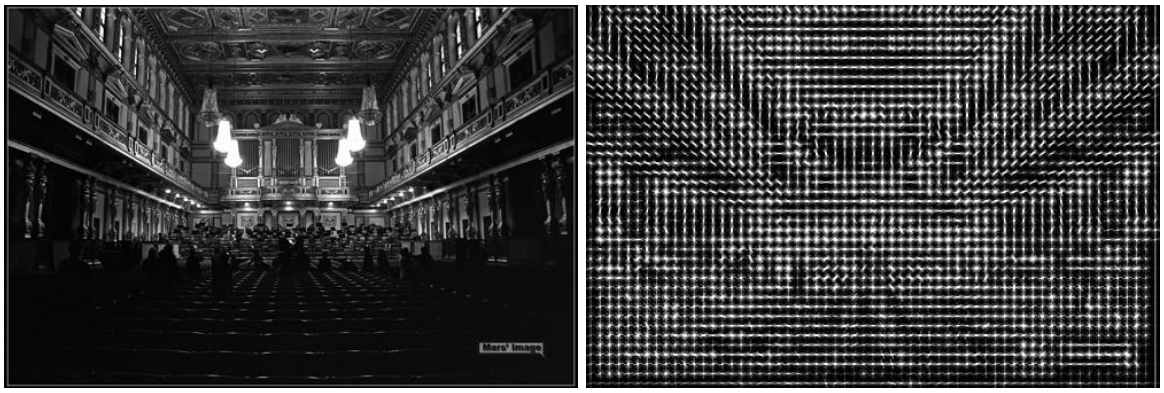
Figure 4.2: **An illustration of multiscale detection.**

Viola-Jones detection algorithm employs a technique called multi-scale object detection to deal with scaling [19], which incrementally scales the image after each round of detection. The size of the sliding window remains unchanged after scaling. The output of the detection process is a series of segments of the images that are determined as the OoI. Due to the use of the sliding window and the multi-scale detection, the OoI could be detected several times by the detector, causing the detector to return multiple overlapping segments. A combining algorithm is usually adopted to combine all the overlapped segments into a single segment. Viola-Jones framework is originally proposed using Haar-like features for detection. However, other types of features, such as LBP [16] and HOG [13] could also be employed for detection, depending on the applications. We discuss the feature used in Lockon in next section.

4.2 Hog: Histogram of Oriented Gradients

Lockon adopts a feature descriptor called Histogram of Oriented Gradients [13] (HOG), which is widely used for object detection. HOG utilizes the local intensity gradients of objects to characterize the appearance and the shape of the OoI. Specifically, it computes the histogram of the occurrences of the gradient orientations in local image segments, and encodes the histogram to obtain the feature descriptor. HOG resembles techniques such as edge orientation histograms, scale-invariant feature transform (SIFT) descriptors, and shape contexts. However, a key difference that separates HOG from other approaches is that HOG utilizes a dense grid of uniformly spaced cells for computing the gradient orientations, and employs overlapping local contrast normalization to improve accuracy. Compared with other features (Haar, LBP, and etc.) that are commonly used in conjunction with cascade detector, HOG has several unique advantages. As the the HOG descriptor is calculated based on local cells, it is robust against geometric and photometric transformations, which mainly occurs in larger spatial regions. Moreover, as Dalal and Triggs discovered [13], coarse spatial sampling, fine orientation sampling, and strong local photometric normalization allows HOG to ignore the movements of minor parts of the objects, as long as the object maintains the same orientation. These traits make HOG highly suitable for Lockon to detect various types of objects.

Computing HOG starts with calculating the gradient. The entire image is first divided into small spatial regions called cells, which can be either rectangle or radial. The gradients and their orientations are then calculated in each cell. In order to account for contrast and brightness changes caused by illumination and shadowing, each cell is locally normalized within a block, which consists of multiple adjacent cells. As the blocks are usually overlapping with each other, each cell could be included by multiple blocks. The blocks could either be rectangular (R-HOG block) which can be considered as square grids, or circular (C-HOG block) which resembles the scale-invariant feature transform descriptors (SIFT). The HOG descriptor is then constructed as the vector of all the components of the normalized histogram of the block. The constructed descriptors can be used as features by object detectors.



(a) Original image. (b) Visualized HOG features of the image.

Figure 4.3: **Visualization of HOG features.**

To help better understand HOG features, we visualize the HOG features using a feature visualization algorithm [1] developed by an MIT group. Fig. 4.3 (a) and (b) show the original image and the visualized HOG features of that image. Fig. 4.3 (b) depicts the dense grids of HOG, and the intensity and direction of the gradients of each grid. We can see that HOG can accurately capture the shape of the objects. Moreover, it can also capture the fine structure of the object even it's poorly illuminated (too strong or too weak), thanks to the local normalization.

4.3 Performance Optimization

Achieving good performance on Viola-Jones cascade detectors heavily relies on the implementation and the parameter tuning of the detector. There are some general optimization guidelines of implementation and tuning for improving detection performance. Specifically, Lockon employs the following optimization methods.

Lockon adopts multiple detectors for detecting a single object, with each detector handle a limited amount of orientation variations of the object. Cascade detectors are usually sensitive to rotation, especially to out-of-plane rotations that distort the aspect ratio of the object. As a result, using a single detector to handle all possible object orientations would not offer good performance. However, using too many detectors inevitably increases the computational complexity and incurs long delay. To tradeoff, we use 2 detectors to handle different perspectives of each OoI. Interestingly, we find that the overall computational complexity does not increase drastically. This

is because using multiple detectors decreases the orientation variations that each detector has to handle, which results in simpler detectors with lower computational complexity.

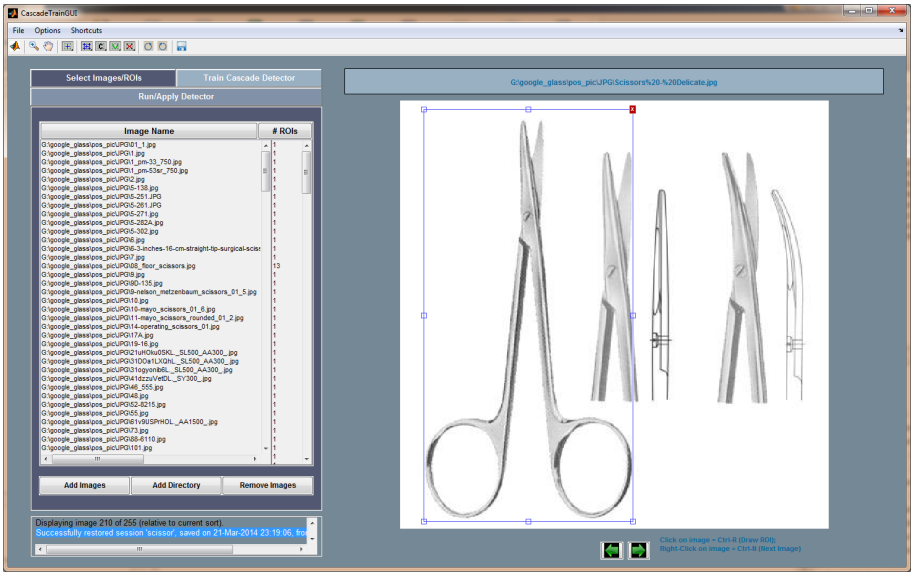


Figure 4.4: **Cascade detectors training GUI provided by Matlab.**

There is a design choice on the number of stages in a detector. Although both of them can offer similar overall false and true positive rates if well trained, the computational complexity of the system with more stages is generally lower than that of the system with less stages. This is because the overall false positive rate decreases exponentially with each additional stage. For instance, given per stage false positive rate 50%, the overall false positive rates of a two-stage detector and a three-stage detector are 25% and 12.5%, respectively. However, the two-stage detector would be significantly more complex than the three-stage detector. Nevertheless, more training data is required for a higher number of stages. Lockon maximizes the number of stages while ensuring that the detector could be trained sufficiently with the amount of available training data.

4.4 Detector Training

We employ the Computer Vision System Toolbox of MATLAB to train the cascade object detector, which offers a user-friendly training GUI, as shown in Fig. 4.4. We implemented 2 detectors for

detecting the wrench sockets (see Fig. 4.5) used in mechanical work in both horizontal and vertical directions. For each detector, a set of positive samples containing 800 images, and a set of negative samples containing 9000 images are supplied to the training function. The negative images has a diverse set of content, which represent the common background that the OoI may appear in. We use the Cascade Training GUI provided by Matlab to mark all the positive objects. The training algorithm follows the standard Viola-Jones algorithm. The negative samples are automatically pulled by the algorithm to each training stage. We perform training on a PC equipped with an Intel i7 2600K CPU and 12G RAM. Training a 20 stages detector with overall false positive rate of 2×10^{-5} and true positive rate of 95% takes about 1.5 hour.

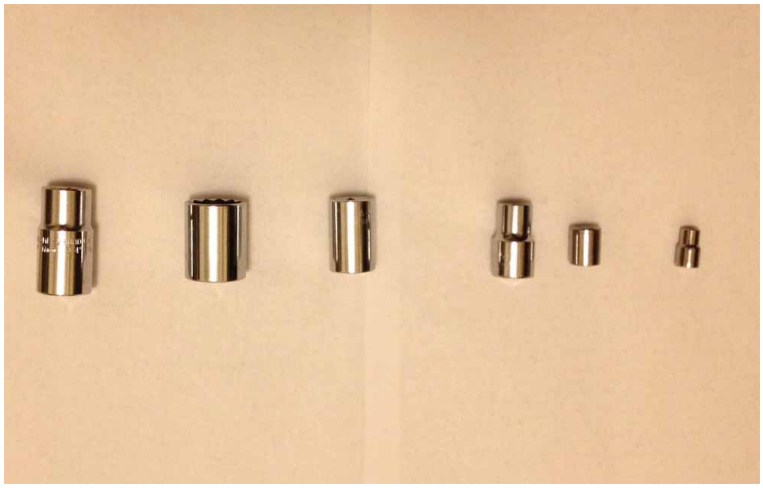


Figure 4.5: **Training Samples.**

CHAPTER 5

MOTION TRIGGERED DETECTION

Compared with other commonly used object detection approaches, the Viola-Jones framework adopted in Lockon is more computationally efficient. However, it still requires significant computational resource to process the images captured by the camera at 30 Hz. Leaving the object detection process running would rapidly drain the battery of the smart glasses. For example, face detection algorithms implemented using OpenCV library drains the battery of a fully charged Google Glass in merely 38 minutes [17]. Moreover, as the presence of the OoI in the camera view is usually considered as a rare event, activating object detection constantly is in fact unnecessary. As a result, object detection process should be turned off when it is not being used by user to conserve the energy of the smart glasses. Lockon implements a trigger to control the activation of the object detection process, which is deactivated by default upon startup.

There are several ways to design the trigger. For example, Lockon can implement UI component such as a menu or a button, which allows the user to activate the object detection process manually. Lockon can also utilize the physical input components (e.g., physical buttons on the phone) on smart glasses to control the activation. However, these methods could work well only on smart devices like smartphones and tablets that offer user-friendly interfaces. For smart glasses that are diminutive and usually lack such interfaces, controlling the activation of the detection process becomes cumbersome. Moreover, for applications that require extensive use of user's hands, such as surgery, it is often impossible for user to manually operate the smart glasses using hands. Lockon can also adopt voice commands issued by the user to activate the detection process. However, the effectiveness of the voice command relies on many factors, including the sensitivity of the microphone, the level of background noise, and the design of the audio signal processing circuit, which vary significantly cross platforms, applications, and environments. As a result, the accuracy of voice command recognition varies significantly and cannot be assured for all scenarios.

Furthermore, in some applications like medical care, the users are often required to keep quiet.

Clear images without significant blur is often required to properly recognize an object. To obtain clear images, the camera on the smart glasses must be staying still when capturing the image. Moreover, it is natural for human to stay still while recognizing an object. This motivates us to design a motion trigger that activates the object detection process intelligently. Specifically, the object detection process is only activated when the smart glasses have been staying still for a certain amount of time. To determine if the smart glasses are still, Lockon employs the accelerometer that is available on most smart glasses to detect the motion. Acceleromters measure acceleration of the smart glasses along three axels. Measurements from the accelerator that are sufficiently close to zero indicate that the smart glasses are still[1]. However, acceleromters implemented on smart glasses usually measure the *proper acceleration*[7] of the device, which is not exactly the rate of velocity change, i.e., the *linear acceleration*. Instead, it is the acceleration associated with the phenomenon of weight experienced by the accelerometer. As result, the acceleration measurements when the device is staying still contain the components of the gravity along one or more axles. Due to this reason, to properly detect if the device is still, the gravity components must be removed from the accelerometer measurements.

5.1 Gravity Removal

A few methods are available to remove gravity from acceleromter measurements and compute the linear acceleration. The first type of methods utilize the gyroscope or compass to measure or estimate the direction of the gravity, using which the linear acceleration could be computed by fusing the acceleromter measurements and the direction of the gravity. However, smart glasses may not be equip with the gyroscope or compass. Moreover, invoking these sensors incurs additional power consumption. The second method, which is employed by Lockon, utilizes a high pass filter to filter out the constant gravity components from the measurements. Specifically, an averaging

[1] Although a zero acceleration could be also caused by the uniform motion of the smart glasses, however, in practice the duration of the uniform motion rarely exceeds a few seconds.

window is adopted to smooth the measurements, and the linear acceleration could be computed by subtracting the smoothed measurement from the instant measurement. This method is less accurate than the sensor fusion approaches and introduces minor delay, however its performance is more than sufficient for Lockon which does not require high motion measurement accuracy or short measurement delay. A pseudo-code for the filter is given in Algorithm 1. Fig. 5.1 depicts the measured linear acceleration of the smart glasses wearing by a user who is trying to take images using the onboard camera. There are two periods (0 s to 2.3 s, and 7.5 s to 10 s) that the device is still when the user takes images. During the time between the two periods, the device is moving by the user who is adjusting the viewing perspective of the camera. It can be seen that the measured acceleration matches the motion of the device very well. This clearly illustrates the effectiveness of the motion trigger.

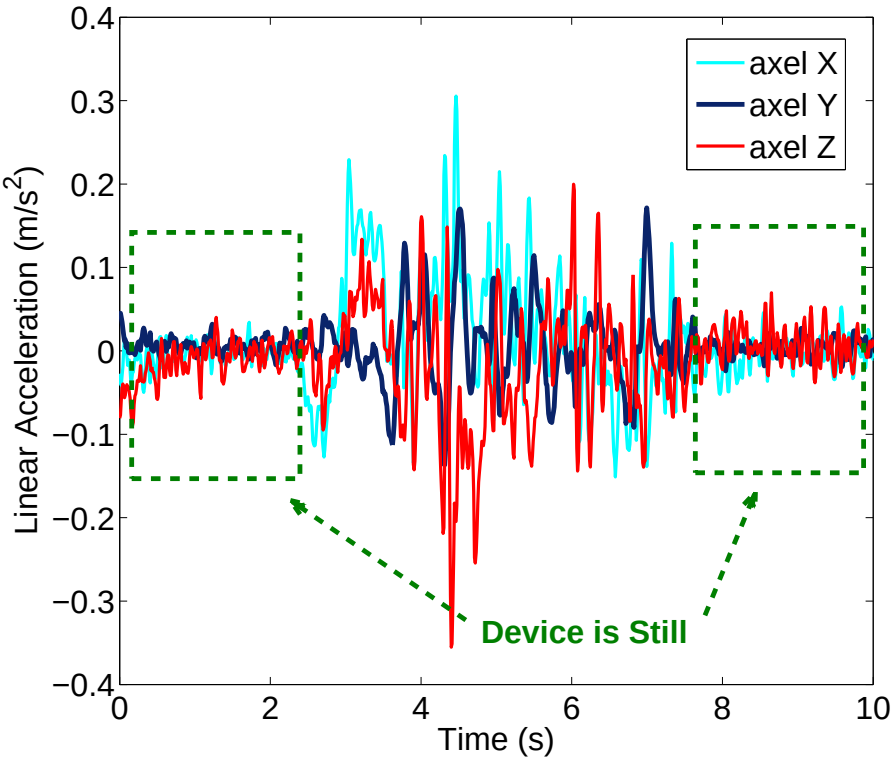


Figure 5.1: **Linear acceleration measurement of a device in motion.**

Lockon continuously monitors the linear acceleration after startup. The object detection process is only activated if the monitored the linear acceleration is constantly below $0.1 m/s^2$ for 1s.

The detection is immediately deactivated if any linear acceleration measurement is above $0.1 m/s^2$.

The pseudo-code of the motion trigger is given in Algorithm 2.

Algorithm 1 *High pass filter for removing gravity*

Input: a_p : proper acceleration samples of a single axel; N: number of proper acceleration samples in buffer; w: length of the window for computing gravity
Output: a_l: computed linear acceleration for the axel.

for all $i \in (1,N)$ **do**
  $g = mean(a_p(i - w/2 : i + w/2))$
  $a_l(i) = a_p(i) - g$
end for
return a_l

Algorithm 2 *Motion Trigger*

Input: $a_{lx}, a_{ly}, and a_{lz}$: linear acceleration samples of three axels; N: number of acceleration samples in buffer per axel; s: accelerometer sampling rate (Sa/s)
Used sub-function: *ActiveDet*: routine to activate object detection; *DeactiveDet*: routine to deactiveate object detection.

$count = 0$
for all $i \in (1,N)$ **do**
  $a_l = \sqrt{a_{lx}(i)^2 + a_{ly}(i)^2 + a_{lz}(i)^2}$
  if $a_l >= 0.1$ **then**
    $count = 0$
  else
    $count++$
  end if
  if $count > s$ **then**
    $ActiveDet()$
  else
    $DeactiveDet()$
  end if
end for

CHAPTER 6

OBJECT DIMENSION MEASUREMENT

Remotely measuring the dimension of the object of interest is very useful for applications like navigation and tourist guiding. As Lockon can detect the object of interest, it can measure the dimension of the detected object projected to the image sensor of the camera. However, to convert it to the physical dimension of the object, Lockon has to know the distance between the camera on the smart device and the OoI ("camera-object distance").

There are generally two methods to measure the camera-object distance on typical smart devices. The first method relies on the auto-focus function of the camera. To focus on an object, the distance between the lens and the image sensor is adjusted until the image of the object is clearly formed on the image sensor. Autofocus automatically adjusts the position of the lens using miniature motors that can be finely controlled. After the object is properly focused, the camera-object distance can be calculated using the camera focal length, and the distance between the lens and the image sensor. Unfortunately, cameras on smart glasses may not support autofocus. Moreover, for systems equipped with autofocus cameras, the distance between the lens and the image sensor may not be exposed to apps. The second method, which is employed by Lockon, involves with using stereo images to measure the camera-object distance with a technique called stereo triangulation [10]. Specifically, this method utilizes the position disparity of the OoI in images taken at different locations, to calculate the camera-object distance. This approach does not require any specific hardware, thus can work on most smart glasses.

6.1 Stereo Triangulation Based Dimension Measurement

A typical digital camera system is consisted of a lens (or a group of lens), an image sensor, and a housing that blocks unwanted external light. Let the focal length of the camera to be f, and the size of the image sensor to be w by h, then a camera system can be illustrated in Fig. 6.1. Assume

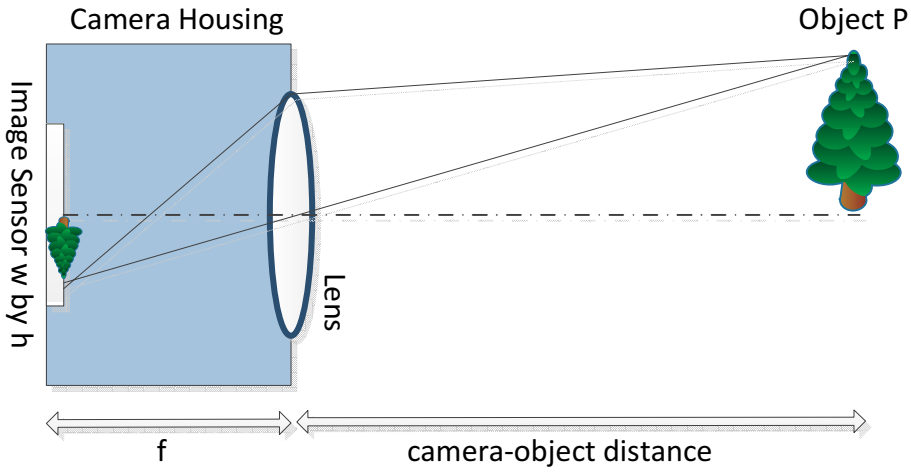


Figure 6.1: **An illustration of typical camera systems.**

that the OoI P is sufficiently far from the camera, then the image plane is roughly positioned at the focal point. In order to perform stereo triangulation, two images containing the OoI must be taken at different positions.

Fig. 6.2 illustrates the scenario when two images are taken at L and R, with the optical axes parallel to each other. The origin of the reference system lies at L, and the distance between L and R is d. Assume that the camera only moves along the X axis. Let x_1 and x_2 be the X coordinates of the images taken at L and R, respectively. Then the Z coordinate of P, i.e., the camera-object distance, could be calculated using simple geometry. Specifically, it is computed as:

$$Z = \frac{df}{x_1 - x_2} \tag{6.1}$$

x_1 and x_2 can be computed from the positions of the OoI in the two images using the following equation:

$$x = (\frac{x_{im}}{N_h} - 0.5)L_h \tag{6.2}$$

where x_{im} is the horizontal pixel index of the OoI in the image, N_h is the horizonal resolution of the image, and L_h is the physical length of the image sensor along the X axis. x_{im} is obtained after Lockon successfully detects the object. d could be estimated by the user, although this would introduce large errors to the measurement of camera-object distance and the dimension of the object. Some fixtures could be used to accurately move the camera by a given distance, which

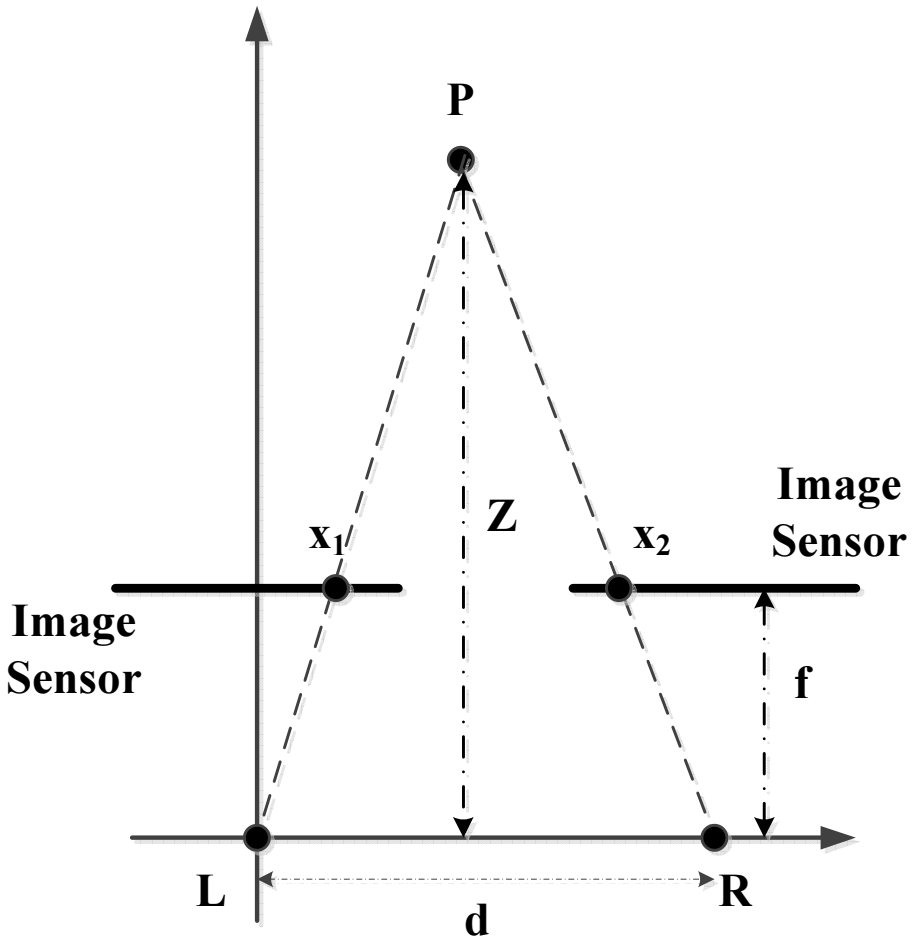


Figure 6.2: **The camera-object distance can be computed using stereo triangulation.**

produces a known d. d could be also estimated from the acceleration measured by the accelerator, although we find that the accelerometers on most smart devices are not sufficiently sensitive to accurately measure d, which incurs large estimation error to the object dimension. We describe a method for generating and measuring a fixed d in the next section.

After Z is obtained, the dimension of the object, D_x and D_y can be estimated using the size of the image segments, represented as dx and dy, using the following equation:

$$D_x = \frac{d_x Z}{f} \tag{6.3}$$

$$D_y = \frac{d_y Z}{f} \tag{6.4}$$

(a) User holds his head upright, facing front when taking the first image.

(b) User tilts his head when taking the second image.

Figure 6.3: **Illustration of head-tilting scheme.**

6.2 Head-tilting Scheme

We designed a scheme called head-tilting for consistently producing d each time the user takes photos and performs stereo triangulation. In this scheme, when taking the first photo, the user holds his head upright, facing front. Before taking the second photo, the user tilts his head toward one shoulder as much as he can, while keeping facing forward. As human can only tilt the head for a certain degree, the resulted d is largely a constant for each user. We also devise a method for Lockon to measure d, using the user's thumb and stereo triangulation. To use this method, the user creates the distance d using the head-tilting method. However, the user also holds his left arm straight and forward, and sticks the thumb up. The thumb should be kept still during the measurement. Lockon approximates the camera-object distance, Z, as the length of the user's arm, which can be estimated by the user's height. Lockon has a built-in thumb detector that reports the locations of the thumb in the two images. d is then calculated using stereo triangulation. This method is illustrated in Fig. 6.3.

The head-tilting method could introduce some errors due to the rotation of the head along Y axis during tilting. This could be compensated using the onboard sensors such as gyroscope and compass that can provide angular movement information. However, this is left for future work. Moreover, we show that in Section 8.2, even without such compensation, our scheme can still

achieve a mean estimation error of only 3.3% when the camera-object distance is smaller than 1 m.

6.3 Discussion

Currently Lockon can only conduct dimension measurement when there is a single OoI in the view, since Lockon cannot differentiate multiple OoIs at different locations and associate OoIs in two images. When there are multiple OoIs presented in the view of the smart glasses, the dimension measurement function would not work. The dimension measurement of multiple objects can be enabled by adding an object tracking function to Lockon, which allows Lockon to track the OoIs during head-tiling. However, the implementation of this function is left for future work.

On some smart devices that equip stereo cameras (two cameras that face the same direction) such as HTC One (M8) [4], the dimension of objects can be measured without moving the camera. Specifically, the two camera can take images simultaneously and the stereo triangulation can be performed using the two images. In such case, the d is a constant value, and the optic axels of the cameras are always parallel. This results in a highly accurate camera-object distance and dimension measurements. Unfortunately, we have yet seen such smart glasses appear.

CHAPTER 7

IMPLEMENTATION

We implemented Lockon on Google Glass Explorer Edition running Android 4.0.3 . We installed OpenCV 2.4.8 library on Google Glass. Lockon is implemented as a standard Android application, which is written in Java.

During the initialization phase, Lockon first loads the trained detectors (.xml files) from local folders specified in its configuration file. To detect additional types of objects other than the two built-in detectors (socket), user can train additional detectors and load them to Lockon. User can also specify which detectors should be loaded from the local detector collection in the configuration file. An accelerometer callback routine is registered to Android OS, which processes the accelerometer measurements once they are generated. Camera and accelerometer are activated in the end of the initialization. The camera takes images at 30 frame-per-second rate.

The sampling rate of accelerometer is configured to 50 Hz to achieve low delay on the motion trigger. To help user determines the status of the motion trigger, Lockon implements a progress bar which is shown on the screen. The length of the bar is proportional to the duration that the device has been staying still since last movement. A full length of the progress bar indicates that the device has been staying still constantly for at least 1 s, and the object detection is ongoing.

When the object detection process is activated, Lockon retrieves the image from the camera buffer once a frame is captured. As the cascade detector implementation in OpenCV only accepts gray-scale image, the retrieved image is converted it to a gray-scale image with 8 bit depth. The commonly used image equalization process is omitted in Lockon, since the HOG feature adopted by Lockon performs normalization on the image locally. A collection of image segments that are determined to contain the OoI are returned by the detectors. The segments that contain the same object are combined and labeled. The processed image with the bounding box indicating the OoI is then displayed on the screen.

The dimension measurement function is automatically invoked after the object detection is activated. Lockon assumes that the initial detection is done with the head of the user upright. After the object is initially detected, its location on the image is recorded. If the user wishes to measure the dimension of the object, he must touch the touchpad on the Glass, which prevents the motion trigger from deactivating the object detection process caused by the following head-tilting. The motion of the device is monitored using accelerometer to determine when the user finishes head-tilting. The detection results before and after head-tilting are feed to the stereo triangulation algorithm to calculate the distance and the dimension of the object. The measurement result is displayed along the bounding box of the object.

We note that Google Glass supports Glassware API that allows the computational intensive task to be conducted on cloud. We will implement part of Lockon with this API in the future. The total ROM footprint of Lockon is about 10 KB, and the RAM usage is about 1 MB.

The training program is implemented using Matlab 2013 and the Computer Vision System Toolbox. We trained two detectors for detecting the wrench sockets used in mechanical work in both horizontal and vertical directions. The sockets we used for training are shown in Fig. 4.5

CHAPTER 8

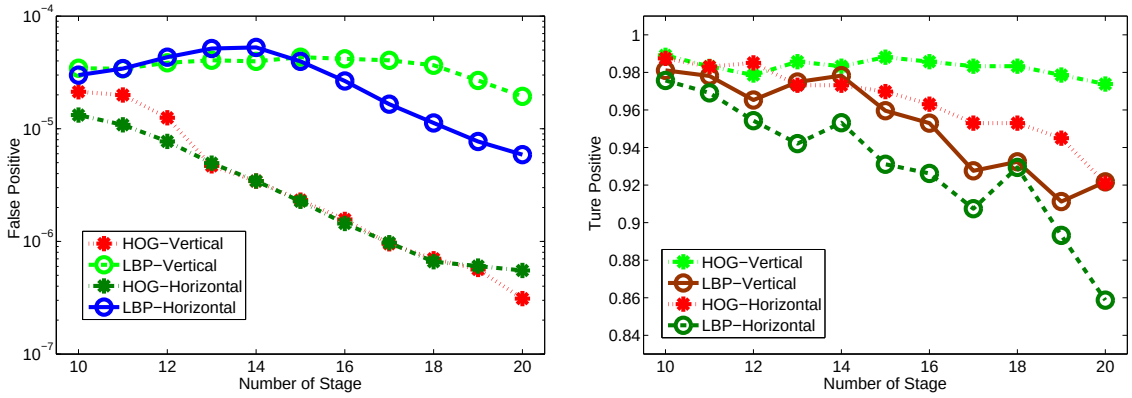
EXPERIMENTATION

8.1 Detection Performance

In this section we evaluate the detection performance of Lockon. We adopt 6 sockets of different sizes as the objects to be detected. We use Google Glass to take about 3000 photos of these sockets lying either horizontally or vertically, at a fixed resolution of 1280 by 720 under different illumination conditions. We randomly pick 1600 photos for training two detectors that handle the two orientations separately. The rest of the photos are left for testing. We use Maltab 2013b and Computer Vision System Toolbox to train the detectors and test the detection accuracy. We set the scaling factor adopted in multiscale detection algorithm to be 1.1.

8.1.1 Detection Accuracy vs Number of Stages

We first evaluate the performance of the object detection subsystem of Lockon. We train the detectors to detect sockets using HOG feature descriptor with number of stages from 10 to 20. We set the per stage detector parameters (false positive rate and true positive rate) the same for all



(a) False positive rates of HOG and LBP detectors vs stages.

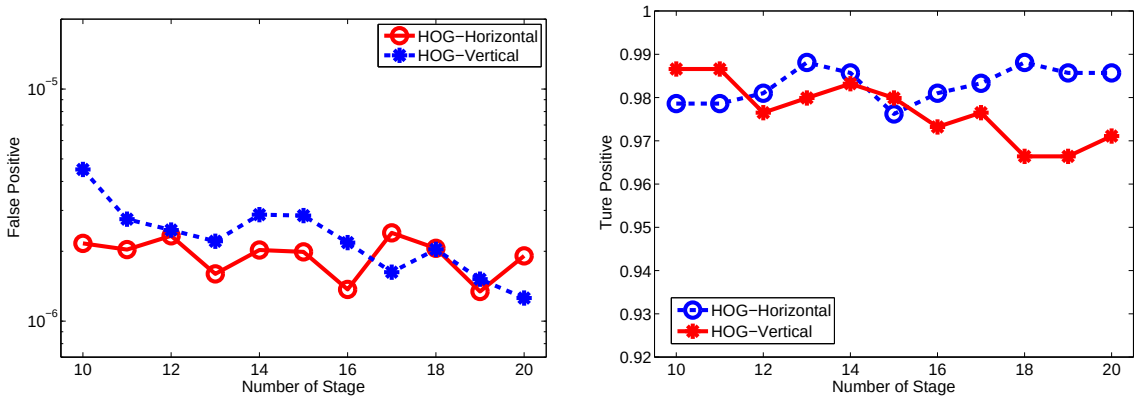
(b) True positive rates of HOG and LBP detectors vs stages.

Figure 8.1: **Detection accuracies of detectors using HOG and LBP feature descriptors.**

detectors. To compare the performance of detectors that use different feature descriptors, we also train detectors using LBP feature descriptor with the same setup. We then test the trained detectors to calculate the overall false positive rate (FP rates) and true positive rate (TP rates) associated with each detector. The false positive rate is calculated as the ratio of the number of negative samples (contain no OoI) that are incorrectly classified as positive (contain OoI), to the total number of samples tested. We calculate the total number of tested samples using the size of the testing image, the size of the sliding window, and the scaling factor. The results are shown in Fig. 8.1.

It can be seen from Fig. 8.1 (a) that the overall FP rates of all curves decrease when the number of stages increases. This confirms our analysis in Section 4.1 that, the overall FP rates decrease exponentially with the increasing of the stages. When the number of stages is 20, the overall FP rates achieved by detectors using HOG and LBP features are 4.3×10^{-7} and 1.26×10^{-5}, respectively. We observe that using HOG feature descriptor incurs significantly lower overall FP rates than using LBP feature descriptor. For example, when the number of stages is 20, detectors using LBP incurs nearly 30X overall FP rate to those using HOG.

From Fig. 8.1 (b) we can observe that the overall TP rates generally decrease when the number of stages increases. This is also consistent with our analysis in Section 4.1. The average maximum and minimum overall TP rates of detectors using HOG, are 0.9883 (10 stages) and 0.9474 (20 stages), respectively. For detectors using LBP, the average maximum and minimum overall TP rates are 0.9784 (10 stages) and 0.8902 (20 stages), respective. Similar to the overall FP rate measurements, detectors using HOG features generally achieve better performance than those using LBP features. We note that such performance difference does not indicate that LBP feature is inferior to HOG feature, as these two feature descriptors favor different applications. When detecting other objects like human faces, it is totally possible that detectors using LBP feature outperform those using HOG feature. However, we do believe for general object detection that Lockon targets, the overall performance of HOG-based detectors would be better than LBP-based detectors.



(a) Measured overall false positive rates vs stages. (b) Measured overall true positive rates vs stages.

Figure 8.2: **Overall false positive rates and true positive rates rates.**

8.1.2 Detection Delay

We evaluate the detection delay of Lockon on Google Glass in this section. We train detectors using HOG feature descriptor with stage numbers from 10 to 20. We adjust the per stage FP rate and TP rate, so that the overall expected FP and TP rates of each detector are equal to 0.00002% and 98%, respectively. To fairly compare the delay incurred by these detectors, we have to make sure that these detectors can achieve similar accuracy. We run test using these detectors and plot the results plotted in Fig. 8.2. We can see that all measured FP and TP rates are about 0.00002% and 98%, respectively, although there are some minor variations.

Having verified that all detectors with different number of stages have roughly the same performance, we then load the detectors to Google Glass, and measure the average detection delay of each detector. We timestamp the start time and the finish time of the detection process for each image, and compute the average detection delay of each detector. Fig. 8.3 shows the results. As we can see, the delay largely decreases when the number of stages increases, although the difference is insignificant ($< \%10$). This is because detectors with higher number of stages tend to be simpler at each stage. This finding confirms the effectiveness of our optimization method described in Section 4.3. Moreover, even if there is only 10 stages, our detector can achieve a delay of less than $350ms$ on Google Glass, which converts to a processing rate of roughly $3Hz$. This speed is sufficient for most applications.

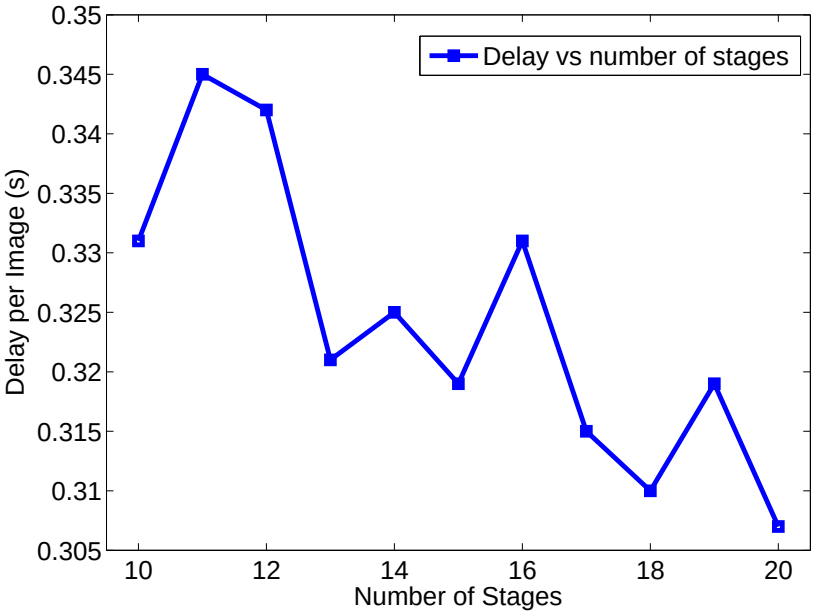
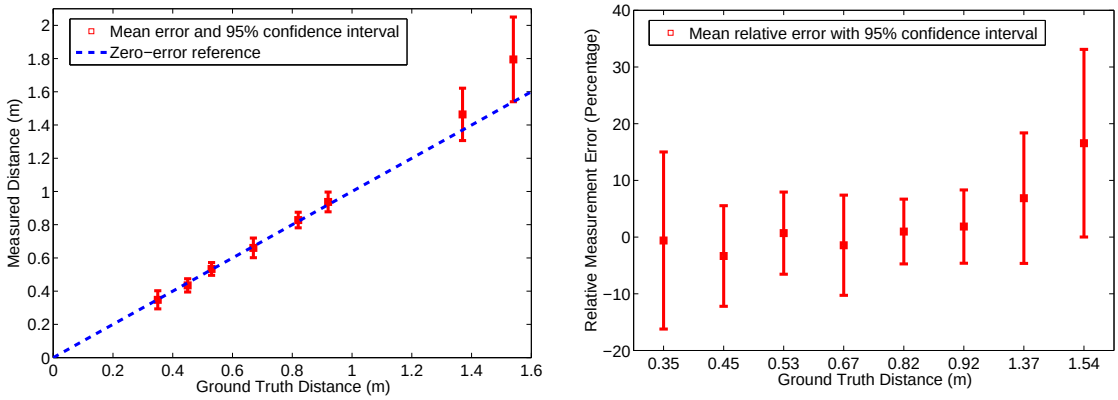


Figure 8.3: **Detection delay incurred on Google Glass.**

It is worth mention that when multiple detectors are activated on the Glass, the detection delay could be significantly increased. This issue could be resolved by running detection process at the cloud side. For example, Google provides a set of API called Glass API for Google Glass to access Google cloud services. To reduce the delay caused by uploading the image to the cloud, the image should be processed locally to extract HOG feature descriptors. Feature compressing techniques [12] can be adopted to further reduce the bandwidth required for transmitting the image. After classification is finished on the cloud, the result is downloaded to the Glass. However, implementing Lockon using cloud is left for future work.

8.2 Accuracy of Distance Measurement

The accuracy of the object dimension measurement largely depends on the accuracy of the camera-object distance measurement. We investigate the error of the camera-object distance measurement in this section. We install Lockon on a Google Glass Explorer Edition, and ask a user to wear the Glass and conduct the experiment. The user uses the head-tilting scheme to measure the distance between the Glass and a socket hanging on the wall, and moves further from the socket after each



(a) Measurement distribution of the camera-object distance measurements.

(b) Relative error of the camera-object distance measurements.

Figure 8.4: **Measurement error of the camera-object distance.**

round of measurements. At each camera-object distance, 10 measurements are taken and recorded. We then compute the errors associated with each camera-object distance, and plot them in Fig. 8.4. We also compute the 95% confidence interval to show the variations of measurements.

Fig. 8.4 (a) shows the measurements at each distance with 95% confidence interval. It can be seen that, when the camera-object distance is smaller than 1 m, Lockon can achieve good measurement accuracy. This observation is confirmed in Fig. 8.4 (b), which shows the relative error associated with each distance. Lockon achieves a maximum mean error of only 3.3% when the camera-object distance is smaller than 1 m. Moreover, 95% of the errors fall below 15%. These errors are the result of head-tilting. Since it is impossible for the user to execute head-tilting exactly the same each time, it introduces small variations to d, i.e., the distance between the positions where the two images are taken. This introduces error to the distance estimation. Furthermore, head-tilting may also cause non-parallel optic axes of the camera when taking images, which also generates error to the measurement, especially at longer distances. Nevertheless, for estimating the distance and dimension of close objects ($<$ 1 m), the accuracy of Lockon is sufficient.

We also observe large errors when the camera-object distance is larger than 1 m. For example, at 1.54 m, the mean error is about 16.56% with a high variance (20%). This is because at longer distance, the measurement becomes much more sensitive to the non-parallel optic axes. A small rotation of the optic axis can generate large error when the distance is far. A possible solution to

this issue is to use sensors like compass to measure and compensate the rotation. However, this function is left for future work.

CHAPTER 9

CONCLUSION

We design and implement a novel object detection and measurement system called Lockon for smart glasses. Lockon takes the advantages of the mounting position of the smart glasses, and provides users two useful functions that can benefit a wide range of applications. Lockon can accurately locate the object of interest (OoI) in the view of the user and inform the user the position of the OoI in real-time, using the front-facing camera equipped on the smart glasses and advanced computer vision and image processing techniques. To conserve energy, Lockon implements a motion trigger to intelligently activate the object detection process only when it is necessary. Lockon can also accurately measure the dimension of the object, with a 3D ranging technique. This capability allows user to remotely estimate the dimension of the object. We implement Lockon on Google Glass Explorer Edition, and evaluate the performance of Lockon using extensive experiments. Our results indicate that Lockon can achieve high detection accuracy (0.95 true positive rate and 4.3×10^{-7} false positive rate), low object dimension measurement error (3.3% when distance is less than 1 m), and low delay (300 ms).

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Hoggles: Visualizing object detection features. http://web.mit.edu/vondrick/ihog, 2013.

[2] Smart glasses market prospects 2013-2018. http://www.juniperresearch.com/reports/smart$_g$lasses, 2013.

[3] Google glass. http://www.google.com/glass/start/, 2014.

[4] Htc one (m8) product page. http://www.htc.com/us/smartphones/htc-one-m8/, 2014.

[5] Lootel. http://www.looktel.com/, 2014.

[6] Nikon d90 product page. http://imaging.nikon.com/lineup/microsite/d90/en/advanced-function/, 2014.

[7] Proper acceleration. http://en.wikipedia.org/wiki/Proper$_a$cceleration, 2014.

[8] Remote sensing wikipedia. http://en.wikipedia.org/wiki/Remote$_s$ensing, 2014.

[9] Runkeeper. http://runkeeper.com/, 2014.

[10] Stereo triangulation wikipedia. http://en.wikipedia.org/wiki/Triangulation$_($computer$_v$ision), 2014.

[11] Wearable devices. http://www.wearabledevices.com/what-is-a-wearable-device/, 2014.

[12] V. Chandrasekhar, G. Takacs, D. Chen, S. Tsai, R. Grzeszczuk, and B. Girod. Chog: Compressed histogram of gradients a low bit-rate feature descriptor. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2504–2511, June 2009.

[13] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893 vol. 1, June 2005.

[14] D. M. Gavrila and V. Philomin. Real-time object detection for ąřsmartąś vehicles. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 87–93. IEEE, 1999.

[15] R. T. Jerome Friedman, Trevor Hastie. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 1998.

[16] T. Ojala, M. Pietikainen, and D. Harwood. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th*

IAPR International Conference on, volume 1, pages 582–585. IEEE, 1994.

[17] L. Z. Robert LiKamWa; Zhen Wang; Aaron Corroll;Felix Lin. Draining our glass: An energy and heat characterization of google glass. Technical Report WUCSE-2003-06, Rice University, 2014.

[18] A. Roy, S. Shinde, and K.-D. Kang. An approach for efficient real time moving object detection. In *ESA*, pages 157–162, 2010.

[19] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511–I–518 vol.1, 2001.

[20] T.-H. Wang, C.-C. Hsu, C.-C. Chen, C.-W. Huang, and Y.-C. Lu. Three-dimensional measurement of a remote object with a single ccd camera. In *Autonomous Robots and Agents, 2009. ICARA 2009. 4th International Conference on*, pages 187–192. IEEE, 2009.

[21] H. Yano, Y. Miyamoto, and H. Iwata. Haptic interface for perceiving remote object using a laser range finder. In *EuroHaptics conference, 2009 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics 2009. Third Joint*, pages 196–201. IEEE, 2009.