Feasibility of Large-Area Photovoltaic Sheets as Indoor Interactive Surfaces

Cristian Sorescu
879091

Submitted to Swansea University in fulfilment
of the requirements for the Degree of Master of Science

Swansea University
Prifysgol Abertawe

Department of Computer Science
Swansea University

May 8, 2020

Declaration

This work has not been previously accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed Cristian Sorescu ..... (candidate)

Date 08/05/2020 .....

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed Cristian Sorescu ..... (candidate)

Date 08/05/2020 .....

Statement 2

I hereby give my consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed Cristian Sorescu ..... (candidate)

Date 08/05/2020 .....

I would like to dedicate this work to science for a sustainable future.
Also, to my friends and family.

Abstract

Human activity recognition in indoor environments is useful for comfortable and efficient living and working in smart homes and buildings. Energy harvesting technologies such as the photovoltaics could offer advantages for low-cost installation, maintenance, portability and energy savings. In this work, we explore the feasibility of large-area indoor photovoltaic (PV) sheets for both energy harvesting and gesture recognition.

To study the feasibility of the PV sheet for gesture recognition, we crafted a prototype with a simpler structure for the sensor and electronics in comparison to related literature. The prototype consists of a large-area PV sheet (110mm width by 28mm height), a maximum power-point tracking (MPPT) module and a Lithium Polymer (LiPo) battery. Numerous hand gestures were performed above PV sheet by partially shadowing the PV cells. These shadows introduce anomalies in the photocurrent signal. Gesture pattern classification and recognition are performed on the photocurrent signals generated by the crafted prototype using machine and deep learning classifiers.

Experimental results show that with the proposed prototype it is possible to detect six distinct hand gestures with average overall accuracy of *86.1%*. The best performing classifier, Random Forest, has achieved *97.2%* overall accuracy results on testing data.

Acknowledgements

Enormous thank you to my academic mentor, Dr Deepak Ranjan Sahoo, and Dr Yogesh Kumar Meena for helping me throughout this project. Also, I would like to thank Dr Michael Edwards for the invaluable advice. As well, thanks to everyone who shared their views and opinions on this project. Last but not least, big thank you to my family and friends who believed in me and supported me throughout my studies.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Today, we are part of a culture where technology plays an important role in almost everything. As technology is evolving, we have smaller and more powerful personal devices consuming lesser power. In recent years, with the introduction of Internet of Things (IoT) and smart devices in the market, we witness a persistent integration of computing into our everyday life such as in homes and offices. The prohibiting factors in the adoption of these technologies is the associated cost of the device and its installation and maintenance and lack of human factors considerations. There are not many self-powered devices in households nowadays. Electricity is required to power most of them, this is increasing our ecological footprint. However, indoor light energy can be captured and reused with photovoltaic (PV) cells. Such energy harvesting devices are enough to power small IoT devices [1, 2].

Interaction with IoT and smart devices with hand gesture recognition is an interesting proposition. Depth cameras and biomechanical sensors are currently used in indoors and personal devices for our comfort, efficient and safe living [3,4]. Having smart homes and buildings recognise and understand the human body language would greatly benefit the users.

Moreover, technological advances in both the hardware and the software fields, provide us with a great number of opportunities to solve problems that could not be solved by computers previously. With the gradual integration of deep learning [5–8] in computer science we observe breakthroughs in solving problems that puzzled machine and deep learning communities for decades [7]. Continuous improvements, promising progress and open access of efficient machine and deep learning libraries such as Tensorflow [9] and Scikit-learn [10] allowed researchers to explore new ways of solving existing science problems. Numerous papers [11–14] are showing the potential of machine and deep learning for gesture recognition. Also, the ease

in the portability of machine and deep learning models provide the opportunity of distributing the computation. Models can be trained on powerful supercomputers and transferred to inferior single board microcontrollers.

In this work, we are exploring the possibility of using a light energy harvester for hand gestures recognition for integration in smart homes and buildings. We specifically aim to examine whether a low cost and portable self-powered PV interactive tabletop is feasible and usable in indoor lighting conditions such as the existing lights in the ceiling for gesture recognition.

1.1 Motivations

Nowadays, It is very common for a household to have a variety of autonomous devices such as smart appliances[1] and remotely controlled equipment such as smart-lights[2] and smart-door-locks[3]. These technologies sparked a great interest in the scientific community to develop ways for a human to smart devices interactions. Furthermore, there is a huge interest in interactive tabletops and a great number of ideas are being presented annually at ACM Interactive Surfaces and Spaces (ISS) conferences. Most of the existing interactive tabletops are sensing gestures by means of touch. A notable example of interactive tabletops is the Microsoft PixelSense (formerly called Microsoft Surface) [4]. There are multiple versions of this product and Microsoft is continuously improving it. Another promising project has been published by Khalilbeigi et al. [15] addressing the challenges of occlusion created by physical objects on interactive tabletops.

Human Activity Detection

There are already multiple applications for human-computer interactions, such as touch screens in smartphones and voice assistants. Another approach that is becoming more popular recently is gesture recognition technology. Research is being conducted every day in this field with promising results. A very popular application of gesture recognition has been introduced with Xbox Kinect [3] in 2010 by Microsoft. More recently, on October 24th 2019, Google has announced a new smartphone which has gesture recognition technology described in their project code-named soli [4].

[1] Whirlpool Smart Appliances. http://www.whirlpool.com/smart-appliances/
[2] Multicolored LED Wi-Fi light. https://uk.lifx.com
[3] August Smart Lock Pro. https://august.com/products/august-smart-lock-pro-connect
[4] MS Surface. http://www.vassigh.com/amv/projects/microsoft-surface/

Self-Powered Devices

Electricity is an important resource in our society today. It is required to power most of the smart devices nowadays. However, the way electricity is produced now is increasing our ecological footprint. Self-powered devices present a great opportunity for a sustainable future. They provide the same functionality whilst also generating their own energy for their needs using technologies described below in Energy Harvesting Technologies. There are numerous projects presented in existing literature [16–18] that have explored and implemented self-powered devices. Their results present promising results and indicate that we are moving in the right direction of sustainable development.

Energy Harvesting Technologies

The aforementioned self-powered devices have additionally spiked interest in energy harvesting technologies. We need to research existing technologies and improve them in order to achieve the set goal of self-powered devices. Energy harvesting devices, such as photovoltaics (PV), provide new opportunities for technological advances. However, there are limitations in terms of the associated cost of the device and its installation and maintenance. Notable examples of energy harvesting devices are presented in [12, 14, 16, 19].

It is obvious that gesture recognition will be widely used in the coming years. Gestures are an important part of interactions between people in our daily lives. Thus, having existing technologies such as smart homes recognise and understand the human body language would greatly benefit the users. We have seen in movies such as Iron Man[5] full workstations where a virtual object is moved in the virtual space by a swipe of a hand. At some point, it seemed like fiction, but we are not far from making such workstations real.

1.1.1 Objectives

The main objectives of this project are:

- **Interactive Tabletop Prototype**

  Present the possibility of an interactive tabletop prototype by integrating a large-area, thin and flexible PV module with off-the-shelf components and a single board computer.

[5] Iron Man Movie. https://www.imdb.com/title/tt0371746/

- **Set of Shadow Based Gestures**

  Create a preliminary set of shadow based gestures in a pilot study in order to capture data for pattern recognition.

- **Comparison of Pattern Matching Algorithms**

  Implement and test a variety of pattern matching algorithms to detect the gestures.

- **Test Application**

  Develop an application with custom toggles, sliders and dial input features and evaluate PV tabletop system for gesture-based interaction.

1.2 Overview

The remainder of Chapter 1 outlines the document structure, the key contributions of this work and an example use-case scenario. Chapter 2 presents literature review, where similarities and differences with the presented project are discussed. In Chapter 3 I will be presenting the approach that was used to solve the set task. Chapter 4 presents the achieved results of the proposed methodology in Chapter 3 and their evaluation. Limitations of the presented system are also discussed in Chapter 4. Lastly, in Chapter 6 main contributions and key points are summarised and the plan for future work is presented.

1.3 Contributions

The main contributions of this work can be seen as follows:

- **Concept of Self-Sustained Gesture Recognition System**

  The concept of a self-sustained gesture recognition system, using the harvested energy from a PV module for near-range hand gesture sensing.

- **Dataset of Gestures Samples**

  Samples of different users gesture readings from the user study will be published with the paper.

- **Gesture Recognition Models**

  Algorithms from conducted experiments demonstrating the gesture recognition abilities, the accuracy of gesture detection and energy harvesting and consumption of the system.

1.4 An Example Use-Case

Mike is standing next to the sofa in front of the TV. The coffee table in front of him has an integrated PV sheet for his hand gestures, Figure 3.5, recognition. He makes a fist to engage the options menu and navigates through the menu with the clockwise and counterclockwise gestures. He then activates the TV remote by performing a raise-lower action. Then he swipes to select the volume option. Lastly, he performs clockwise or counterclockwise hand motions to adjust the volume setting as seen in Figure 1.1. He then makes a fist and with clockwise or counterclockwise hand motions selects the remote control of the smart light and changes its colour.

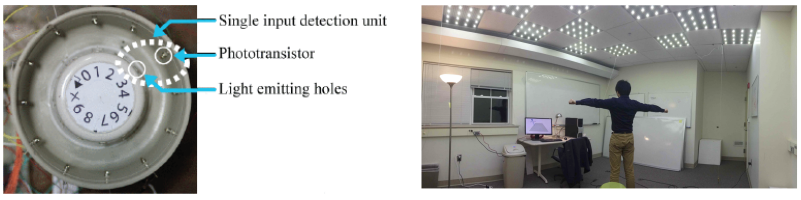
Figure 1.1: An Example Use-Case. Mike is watching TV. The coffee table in front of him has an integrated PV sheet for hand gesture recognition which controls the TV. Mike is performing the clockwise gesture to adjust the volume.

Chapter 2

Related Work

Previous work has addressed the use of PV cells for energy harvesting in outdoor environments. For example, PV Glasses use semi-transparent organic PV cells as lenses to harvest light energy to power the ultra-low power microelectronic circuit and displays [16]. PV materials have been combined with digital displays to present a prototype of ultra low-power displays [20]. Amaravati et al. presented an ultra low-power smart camera for gesture detection and powered it by ambient light harvested through PV cells [21]. Kartsch et al. presented a novel fully-flexible wearable EMG gesture recognition device and powered by an ultra-thin PV cell [12]. Yu et al. discussed a self-powered wearable tactile feedback system, powered by an organic PV module [17]. In contrast, we show both energy harvesting and gesture recognition in indoor settings using the PV cells like in PV-tiles [18], albeit using large-area PV cells.

2.1 Light-Based Interactive Technologies



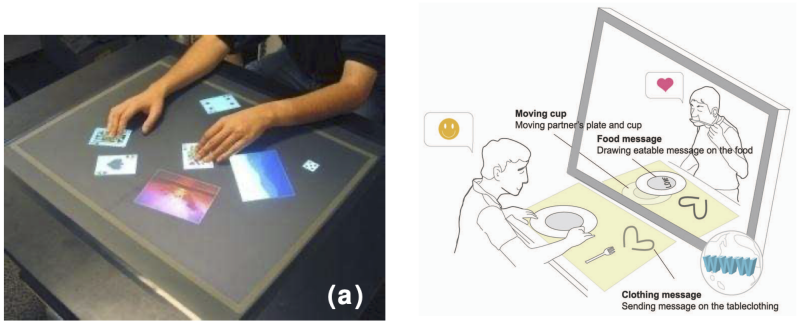
(a) (b)

Figure 2.1: (a) Top view of touchless dial prototype presented in [22], (b) Presented StarLight system is reusing ceiling LED panels together with a few photodiodes to reconstruct a mobile user's skeleton [23].

Interactive technologies provide an engaging interactive experience to people through enriched multi-sensory communication. Chakraborty et al. proposed a visible light-based gesture recognition system and used phototransistors (PT) as a light sensor for gesture detection [22]. They investigate a novel model for recognizing finger and hand gestures based on detecting the line of sight (LOS) and reflection characteristics of the visible light. Their dial prototype is shown in Figure 2.1 (a). They introduced and built a low-cost, limited-resource, and ultra-low-power device for recognizing hand gestures, which does not demand any additional power-emitting source rather than available room light. Our work is very similar, however we use off-the-shelf PV cells instead of complex circuitry that combines multiple phototransistors.

Zhou et al. introduced integrated Visible Light Communication (iVLC) and used modulated LED lights for communication between network devices [24]. They present the VLC concept for both networking and sensing. A photodetector (i.e., a photodiode or non-imaging receiver) and imaging sensor (i.e., camera sensor) are used as VLC receivers for receiving the signal transmitted by an LED luminaire [25]. Li et al. presented StarLight [23], an infrastructure-based sensing system used in a 3.6 m $\times$ 4.8 m office room, with customized 20 LED panels and 20 photodiodes. Example use of StarLight is presented in Figure 2.1 (b). Zhang et al. explored visible light-based device-free localization (DFL) method, which has been widely developed for many applications including gesture recognition [26, 27]. It should be noted that the traditional DFL technique generally uses radio frequency (RF) [28], camera [29], ultrasound [30] or infrared sensors [31] to accomplish these tasks. Amaravati et al. presented an ultralow-power smart camera with gesture detection. By enabling ultralow energy consumption, they demonstrate that the system is powered by ambient light harvested through PV cells whose output is regulated by TI's dc–dc buck converter with maximum power point tracking [21]. Zijie et al. present a newly designed photoelectrochromic smart windows (PECSW) comprising of a largescale electrochromic device (ECD) and several integrated FDSSCs [32]. This device can be directly driven by sunlight and the transmittance of the ECD can be adjusted by the light intensity, which can adjust indoor light intelligently and reduce energy exchange with the outside world. In contrast, we use indoor light sources in the existing infrastructure. Furthermore, we use the captured light to both harvest energy and perform gesture recognition.

2.2 Interactive Surfaces



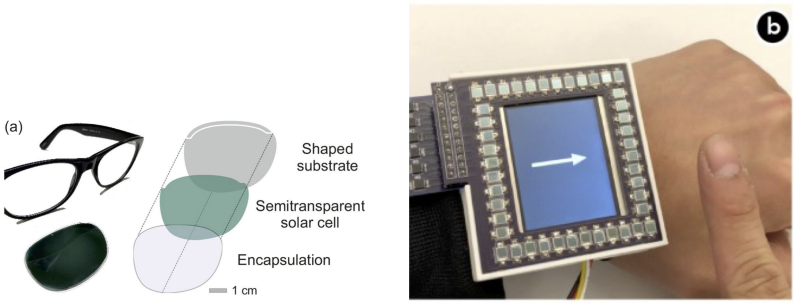
(a) (b)

Figure 2.2: (a) ShapeTouch prototype platform [33], (b) An example use-case scenario of the Codine system presented in [34].

Florian et al. present prototypical implementation of a low-cost, unobtrusive interactive surface, integrated with the dining table in a real-world living room. They incorporated three examples of interactive dining table with and augmented board game, ambient weather notifications, and augmented meal [35, 36]. ShapeTouch [33] explored a natural and intuitive interaction style that directly utilize the contact shape on interactive surfaces to manipulations of objects and interactors. The ShapeTouch platform is depicted in Figure 2.2 (a). Furthermore, CoDine [34] presents a dining table embedded with interactive subsystems that augment and transport the experience of communal family dining to create a sense of coexistence among remote family members. The CoDine system connects multiple people in a different location through shared dining activities. An example use case of the CoDine system is shown in Figure 2.2 (b). SmartSkin [37] introduced sensor architecture for making interactive surfaces that are sensitive to human hand and finger gestures. In particular, the sensor recognizes multiple hand positions and shapes and calculates the distance between the hand and the surface by using capacitive sensing and a mesh-shaped antenna. Mohammadreza Khalilbeigi et al. presented ObjecTop in their paper "ObjecTop: Occlusion Awareness of Physical Objects on Interactive Tabletops" [15]. The authors addressed the challenges of occlusion created by physical objects on interactive tabletops. Their system support tabletop display applications involving both physical and virtual objects. In addition, they conducted an in-depth user study comparing ObjecTop with conventional tabletop interfaces. Their studies results show that occlusion-aware techniques outperform the conventional tabletop interfaces [15]. Dr Deepak Ranjan Sahoo et al. published a paper presenting a tabletop display that provides controlled

self-actuated deformation named TableHop [38]. The system is made of a highly stretchable pure spandex fabric that is actuated using electrodes mounted on its top or underside. The use of transparent indium tin oxide electrodes and high-voltage modulation allowed to produce controlled surface deformations that make the tabletop interactive. Experimental results were presented for the shape of the deformation and frequency of the vibration of the surface [38]. On the other hand, we are presenting a contactless system where gestural input is generated by partially shadowing the PV sheet.

2.3 PV Interactive Devices



(a) (b)

Figure 2.3: (a) Illustration of components used for the solar glasses presented in [16], (b) self-powered smartwatch presented in [1].

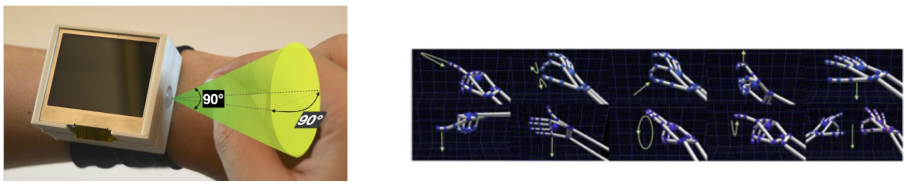
Photodiode and PV modules are becoming more popular for gesture recognition systems. PV, thermoelectric and electromagnetic induction are the well-established technologies for energy harvesting. An example is to use mechanical energy harvesters to both sense mechanical vibration and communicate that information. PV Glasses [16] are solution-processed (low-cost and scalable), semi-transparent organic PV cells that are implemented as lenses in sunglasses. The components used to make the lenses of the PV glasses are illustrated in Figure 2.3 (a). Light energy is harvested by the lenses to provide electrical power to a microelectronic circuit and ultra-low power displays, in a similar way to the interactive PV tiles [18]. Fiber-shaped dye-sensitized PV cells (FDSSCs) has been used to convert the harvested energies into electricity in a self-charging power textile system [19]. The prototype of the system presents a fabric-hybridized self-charging power system not only for harvesting PV energy from ambient light but also for gathering mechanical energy from human motion. Both of the harvested energies can be easily converted into electricity by using fiber-shaped DSSCs (F-DSSCs) (for PV

energy) and fiber-shaped TENGs (F-TENGs) (for mechanical energy) and then further stored as chemical energy in fiber-shaped SCs (F-SCs). Recent work also discusses a self-powered wearable tactile feedback system, consisting of a piezoelectric actuator, driving circuitry, and an OPV module, showing the "bright" future of self-powered active smart textiles [17]. The novelty of our PV system, however, is in its ability to use semi-transparent PV cells to both harvest energy and act as shadow-based gesture detection elements.

Manabe et al. [39] proposed a touch sensing technique using the partial shadowing of a small Si PV module. Li et al. presented LiSense which relies on an array of LEDs installed in the ceiling to reconstruct the entire human skeleton in 3D for gestures recognition [40]. Venkatnarayan et al. presented LiGest, an improvement over LiSense which can sense hand gestures from further distance and is user agnostic, orientation agnostic and lighting condition agnostic [41]. Mahina-Diana Kaholokula presented GestureLite which detects hand gestures using Si PDs. These papers do not discuss energy harvesting.

Li et al. presented self-powered watches, shown in Figure 2.3 (b), and glasses for finger gesture recognition using the ambient light using an array of PDs [1]. LiGest [41] is an improvement over LiSense [40]. LiSense is very similar to LiGest, in terms that it is examining the whole human body for gestures. However, LiSense brings significant improvements. LiSense is more adaptive to environmental changes. As with LiGest, LiSense is scanning the entire body for gestures, while the proposed system is more dedicated to hand sensing. On the contrary, we consider a single PV module with large-area with simpler electronics for large interactive surfaces with hand gesture sensing.

2.4 Machine/Deep Learning Gesture Recognition



(a) (b)

Figure 2.4: (a) Pyro smartwatch prototype [42], (b) 10 hand gestures from the Handicraft-Gesture set examned in [11].

Camera-based approaches are the most common gesture recognition approaches today. However, camera-based systems for gesture recognition are more limited. Furthermore, with approaches using cameras privacy concerns are introduced for their users. Julien Epps et al. published a paper titled "A study of hand shape use in tabletop gesture interaction" [43]. At the time there was little research conducted into observing user preferences for tabletop gesture interaction. Especially the case for computer vision-based gesture input. They strongly believe that having computers recognising different hand shapes would greatly improve human-computer interactions. In the paper, the authors discuss the results of an observational study of manual gesture input for a tabletop display. They suggest the use of different hands shapes for input and the advantage of combined touch screen and computer vision gesture input. In addition, the possibilities for flexible two-handed interaction were discussed as well in [43]. Lu et al. [11] proposed a new feature vector which is used for dynamic hand gestures detection using just a Leap Motion controller. They used a Hidden Conditional Neural Field classifier for dynamic hand gesture recognition and showed an accuracy of 89.5% for LeapMotion-Gesture3D data-set and 95.0% for the Handicraft-Gesture data-set presented in Figure 2.4 (b) [11]. However, just as with the Xbox Kinect [3] the use of a camera is not ideal for hand gestures detection. Moreover, we present gestural input recognition based on analysis of photocurrent obtained from a PV sheet rather than a camera.

As discussed in Section 2.3, the gesture recognition field is moving towards the direction of self-sustainable devices and looking at other approaches such as using PVs. Gong et al. presented Pyro [42] using an infrared pyroelectric sensor that can sense gesture signals using a passive infrared (PIR) light sensor made of pyroelectric materials. Their smartwatch prototype is illustrated in Figure 2.4 (a). They evaluated their system performance with ten participants and got 93.9% cross-validation accuracy and 84.9% leave-one-session-out accuracy on the defined six thumb-tip gestures [42]. Kartsch et al. present a novel fully-flexible wearable device, achieving an unobtrusive self-sustaining smart system for EMG gesture recognition, designed on a flexible wristband and powered by a PV ultra-thin energy cell. Leveraging an array of passive EMG sensors, they designed a smart system capable to recognize directly on-board up to 5 hand gestures with 4 sensors located on the wrist. It achieves an accuracy of 94.02% on the 5 classes/gestures [12]. Ma et al. explored the transparent PV cells for hand gesture sensing and reported gesture recognition accuracy of 94.96% and 94.52% under 500 lux and 2600 lux, respectively [13]. They tested numerous algorithms, such as Support Vector Machine (SVM), Random Forests (RF), K Nearest Neighbor (KNN) and Decision Tree, to

classify the gathered data. Ma et al. also presented a battery-free system which can perform gesture recognition using PV cells by analysing patterns of the photocurrent [14]. They evaluated the system with two types of PV cells, see-through and opaque. In their project, 6,960 gesture samples were analysed defining six distinct gestures. They reported that opaque PV cells show an accuracy of 97.2% and transparent PV cells can achieve accuracy of 96% whilst consuming 44% less power [14]. A a list of classifiers and achieved results discussed above is presented in Table 2.1. In contrast to the papers discussed in this section, we present operation with indoor ambient light only and provide larger surfaces for the user to interact with. Furthermore, we have also taken into consideration convolutional neural networks (CNN) and gradient boosting classifiers.

Classifier	**Accuracy**	**Paper**
KNN	96%	[14]
*HCNF	95%	[11]
RF	94.96%	[13]
SVM	94.02%	[12]
RF	93.9%	[42]

Table 2.1: Presented best accuracies of classifiers considered in related work gesture recognition. Sorted from best to worst. *HCNF: Hidden Conditional Neural Field.

Chapter 3

Approach

In recent years, there was a huge expansion in technological development. Devices are getting smaller and require less power and peripherals to operate. As the field is undergoing continuous changes, new ways of user interactions are needed. For instance, people interact with each other in numerous ways, one is body language. Getting computers to comprehend human body language will allow control of systems to be more straight forward and intuitive. As mentioned in [44], a key factor of a good user interface is familiarity and intuitiveness.

A great number of researchers have already developed fully working gesture recognition interfaces. As for today, the most common method of gesture recognition is to make use of cameras. Cameras take continuous picture and feed them into image processing algorithms. A notable example is the Xbox Kinect [3] which can found in many households. However, camera-based approaches introduce privacy concerns for their users. Other approaches have been used on different platforms throughout the years. A very popular gesture recognition device is the Wii remote. These devices generate valuable data using numerous sensors, such as accelerometers. However, oftentimes it is inconvenient to wear such sensing devices.

Another important factor is sustainability. Electricity is a crucial resource nowadays, it made possible great technological advances and provides light and warmth for everyone around the world. But, the way we obtain this resource is increasing our ecological footprint on the planet and may lead to a not so sustainable future. There is great progress in improving the ways we generate electricity, such as windmills, hydroelectric systems and solar power systems. As stated in Section 1.1 self-powered devices that make use of sustainable energy harvesting technologies provide a great opportunity to build a better future substantially reducing pollution and for greater achievements in science. Notable self-powered devices are presented

in [16–18]. Also, remarkable energy harvesting devices are presented in [12, 14, 16, 19]. Moreover, we need to identify the most effective gesture classifiers for sustainability reasons.

In this project, the software and hardware aspects are equally important. First, we set-up a prototype that allowed us to determine if our idea is feasible. Once it became clear that distinct data can be extracted for different gestures, we defined a preliminary set of gestures. Then we collected multiple readings of photocurrent for each defined gesture. The readings were analysed in terms of anomalies and used for gesture classification. To classify the gestures multiple machine and deep learning classifiers were used in order to determine the most effective and suitable one for this project. The main objective of this work is gesture recognition.

The main challenges with this approach are light conditions. We cannot assume control over the lighting in the environment that the system operates. Different light sources may introduce unpredictable noise generated by the level of light, light direction and flickering. Furthermore, the system must recognise equally the same gestures regardless of the user or orientation of the PV sheet. In the worst-case scenario, the absence of light, the system would fail to operate.

3.1 Prototype

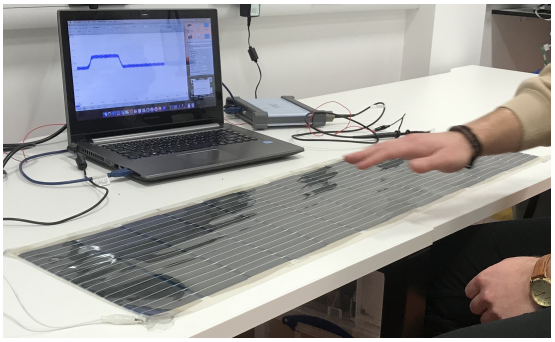


Figure 3.1: Prototype: The PV sheet and the recognition signals are shown.

The current prototype consists of a large-area PV sheet[1], an energy harvesting circuit board and a single-board microcontroller. The experimental setup is shown in Figure 3.1. The length and width of the PV sheet are 110 mm and 28 mm. It can generate up to 2.4 W/m with voltage and current 50-60 V and 60-70 mA, Table 3.1.

[1] infinityPV. https://infinitypv.com/products/opv/solar-tape

Type	Length (mm)	Voltage pr. meter (V)	Current (mA)	Power (mW pr.m)	Cost (€ pr. m)[d]
Bidirectional	110	56-60	60-70	ca. 2400	120 (110)

Table 3.1: infinityPV, Solar Tape General Specifications, 110mm wide bidirectional PV tape (> 4%). d: The quoted value is with lined adhesive on the backside.[1]

In photovoltaics, the light is directly converted into electricity at the atomic level as illustrated in Figure 3.2. The cells are designed to supply electricity at a certain voltage. The generated current is directly dependent on how much light strikes the module [45]. Furthermore, PV modules/sheets can be connected in both series and parallel electrical arrangements [45] (arrays) in order to cover large surfaces.

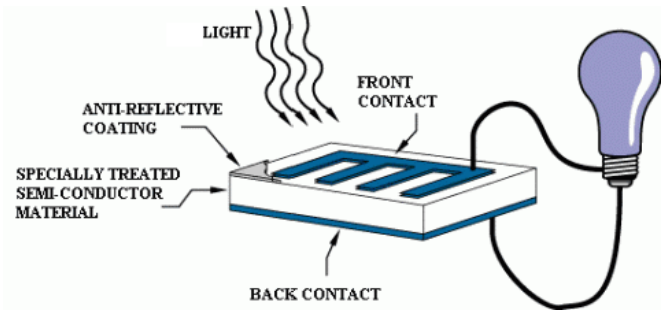


Figure 3.2: Illustrated operation of a basic photovoltaic cell. (edited) [45].

Both, energy harvesting and recognition signals are sampled using the *two* electrodes of the PV sheet only. During energy harvesting, the PV module generates predominantly DC electricity as shown in Figure 4.1 due to stable indoor lighting. The recognition signal appears as AC modulation which is filtered by a series capacitor. Furthermore, the generated electricity can be used to power entirely the system by continuously charging the battery. The energy harvesting circuit board consists of a maximum power-point tracking (MPPT) module which continuously charges a rechargeable Lithium Polymer (LiPo) battery. The gesture recognition circuit board is connected to the load of the MPPT board. Using this prototype will not only greatly lower the costs and used resources, but it will also make the system self-contained and portable to a certain extent. The framework for the above-proposed system is illustrated in Figure 3.3.

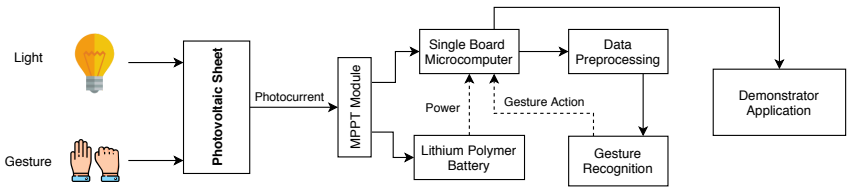


Figure 3.3: Illustrated framework of the proposed prototype for gesture recognition and energy harvesting.

This prototype has many advantages over traditional interactive tabletop devices. Unlike traditional interactive tabletops, it does not use any displays, thus drastically reducing the power consumption. This allows it to be self-contained. Moreover, there are no delays in response time, and it requires little to no setup. A novice user would be able to use the system with minimal effort. Although the prototype does not require a lot of energy to operate, it is still essential that we develop efficient gesture recognition algorithms. Perhaps, one day we could even fully remove the batteries. Many devices with these characteristics have been already developed, such as battery-less cellphones [46] and cameras [47].

3.2 Gesture Recognition

In this section, I will be describing how the gestures were created and analysed. Specifically, I will show how meaningful features were extracted from the photocurrent data. I also discuss how the data is collected. Moreover, I present considered gesture classification approaches and their parameters. Lastly, I show a simple HTML application that is intended for initial testing of the system.

3.2.1 Gestures Set Creation

Usually, the photocurrent data has a straight-line shape on the *xy* axis with insignificant noise. But, as described in Section 3.1, when the light above the PV module is blocked, a certain type of noise is introduced in the photocurrent readings. The shape of the photocurrent data on the *xy* axis changes, thus introducing anomalies. Once the prototype was built, we started the experiments. The conducted experiments allowed us to determine distinct gestures for this project. Numerous gestures, such as swipes and circular hand motions, were considered based on related literature [13, 48]. Initially, the PV sheet was connected to an oscilloscope from the

company Pico Technology[2] in order to acquire meaningful data and have a better understanding of appropriate settings for data collection.

I used the PicoScope 3000 Series that is classified as a USB-powered PC oscilloscope. It can also be plugged into a power outlet. These oscilloscopes are small, light, and portable. These oscilloscopes offer 2 or 4 analog channels and a built-in function / arbitrary waveform generator[2].

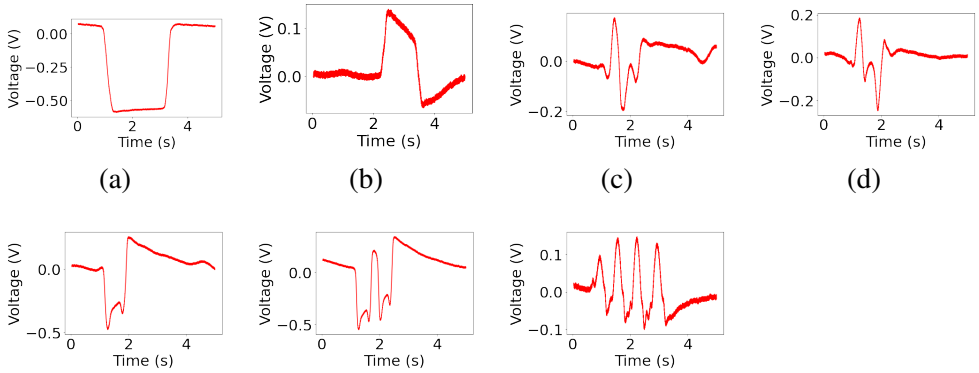
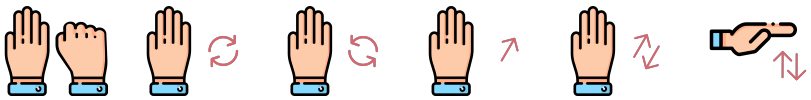


Figure 3.4: Initial Readings of the photocurrent, (a) hand detection, (b) fist (open/close), (c) Clockwise motion, (d) Counterclockwise motion, (e) Singular swipe motion, (f) Continuous swipes motion, and (g) raise/lower hand motion.

Initial readings from the experiments are shown in Figure 3.4. Our readings are univariate time-series data where x axis represents the time and the y axis represents the voltage (V). The time span for each reading is 5 seconds. The voltage is presented as a range, from $\pm 1V$. From the conducted experiments, it was clear that by partially shadowing the PV sheet when performing different hand gestures, such as hand swipes and fist formation, distinct types of noise are encountered.

Based on the experimental results we defined a preliminary set of 6 distinct and distinguishable gestures. The defined gestures are the following, fist formation depicted in Figure 3.5 (a), clockwise hand rotation depicted in Figure 3.5 (b), counterclockwise hand rotation depicted in Figure 3.5 (c), swipe once depicted in Figure 3.5 (d), swipe continuously depicted in Figure 3.5 (e), and hand raise/lower depicted in Figure 3.5 (f).

[2]Pico oscilloscope. https://www.picotech.com/library/oscilloscopes/usage-statistics



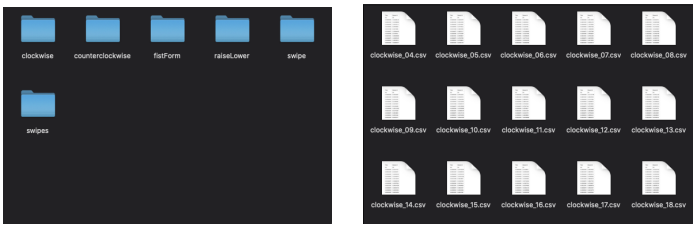
(a) (b) (c) (d) (e) (f)

Figure 3.5: Defined gestures, (a) Fist open/close, (b) clockwise hand rotation, (c) counterclockwise rotation, (d) swipe once, (e) swipe continuously, and (f) hand up/down.

3.2.2 Data Collection

Initially, it was planned to have design sessions where participants would perform gestures from a preliminary set defined in Section 3.2.1. However, due to time constraints and the latest unfortunate turn of events of a global pandemic, the design sessions were cancelled. As a backup, I have a relatively small dataset of gestures samples performed within the research group.

The gestures have been performed in the maker's lab. The lab was well-lit as it has multiple fluorescent lights in the ceiling. To collect the data I used the oscilloscope that was described in Section 3.2.1. Raw data consists of 15004 photocurrent V points across a 5-second time period. This data was acquired at the minimum sampling rate of 50 samples with a collection time of 500 milliseconds per division. The input range of the photocurrent is $\pm 1V$. The resulting readings of the photovoltaic module are univariate time-series data-points of the produced photocurrent as seen in Figure 3.4.



(a) (b)

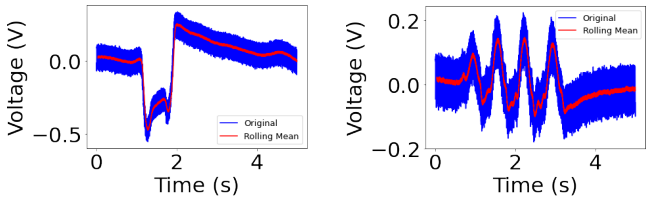
Figure 3.6: Screenshot of the samples folders and files, (a) Folders sorted by gesture name containing numerous samples for each gesture, (b) Example content of the folder containing clockwise gesture samples.

Collected data of performed gestures in the form of voltage readings were saved in Comma-separated values (CSV) files for feature extraction and later for gesture classification. The resulting dataset consists of 154 samples, on average 25 samples per gesture. Each file contains data separated in 2 columns Time (s) and Channel A (V). The total number of rows in each file

is 15007, 3 initial rows are headers rows. Gestures samples have been sorted into folders, see Figure 3.6 (a), and each file has an appropriate name corresponding to the data of which gesture it holds, see Figure 3.6 (b). Sorting the data is particularly useful later on for automation of gesture classification, Section 3.2.4, in terms of automated extraction of class names.

3.2.3 Data Pre-Processing

In order to reduce the classification time of the photocurrent data whilst getting meaningful results, the data-points have to be reduced. For this purpose, we extract features, thus requiring less computational power. But first, the photocurrent data has to be calibrated according to the medium that it operates in.



(a) (b)

Figure 3.7: Depicted original raw signals of the (a) swipe and (b) raiseLower gestures. The thicker line of blue colour is the original signal and the thinner line of colour red is the rolling mean of the original signal.

Calibration: The system should behave the same regardless of the medium it operates in. Because we cannot be assuming control over the light conditions, data is be calibrated appropriately based on light intensity. The system reads the intensity of the photocurrent first, measured in voltage. Then the readings are offset by the maximum value. The calibration is performed on the hand detection shown in Figure 3.4 (a). This technique helps to overcome the inconsistent readings challenge. Furthermore, I am extracting the rolling mean with a window of 100 to reduce the noise significantly whilst preserving the same shape. After the rolling mean is extracted and missing values removed, the size of the gestures sets is reduced to 14905 each. The signals of the swipe and raiseLower gestures before, blue colour, and after, red colour, the application of the rolling mean are depicted in Figure 3.7.

Features Extraction: As mentioned in Section 3.2.1, the readings are represented as univariate sequence of voltage/time data-points, of the generated photocurrent from the PV module.

And the defined gestures are anomalies in the photocurrent signal. Therefore, we analyse the variability in the shapes of produced photocurrent voltage data.

One approach was to feed the data into a convolutional neural network (CNN) model for feature extraction. However, for machine learning approaches, such as random forest (RF) Classifiers, features have to be engineered manually. The advantage of a RF classifier, over (CNN) for instance, is the ease in training and testing. The training can be accomplished in seconds, in comparison to CNNs that can take hours.

First, using the discrete wavelet transform (DWT) from the PyWavelets Python library described in [49] I have split the signal into sub-bands. Sub-bands are simply a list of coefficients values of the given data.

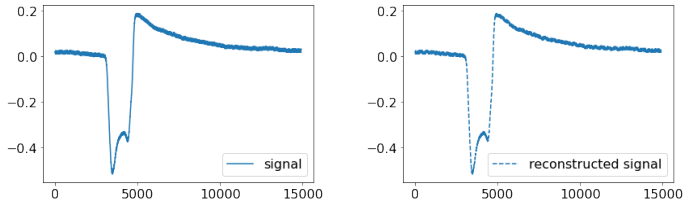
The DWT are used to convolve wavelets on given data to extract meaningful information. Wavelets are mathematical functions that look like the shape of wave oscilations. The wave start at 0 and oscilate to 0 in the end. In mathematics 2 types of functions describe a wavelet, the wavelet function as known as the mother function and the scaling function known as the father function. The wavelet function is denoted as $\psi(t)$ and the scaling function as $\phi(t)$ [50]. These functions have to satisfy certain conditions shown in Equation (3.1).

$$||\psi(t)||^2 = \int |\psi(t)|^2 dt < \infty, \int |\psi(t)| dt < \infty, \int \psi(t) dt = 0, \int \phi(t) dt = 1 \tag{3.1}$$

The DWT allows to extract either a specific amount of sub-bands or as many as possible. To extract the sub-bands I used the first Daubechies (db1) mother wavelet which the same as the Haar wavelet. The Haar wavelet function $\psi_{Haar}(t)$ and scaling function $\phi_{Haar}(t)$ are described in Equation (3.2).

$$\psi_{Haar}(t) = \begin{cases} -1 & n = 0 \\ 1 & n = 1 \\ 0 & \text{otherwise} \end{cases} \qquad \phi_{Haar}(t) = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

After experimenting with different wavelet families, such Daubechies, Biorthogonal and Coiflets, I decided to use the db1 wavelet function. This is because features extracted with orthogonal wavelets performed better in gesture classification than the non-orthogonal ones. Example reconstructed signal of one of the single swipe gesture samples signals is presented in Figure 3.8 (b).



(a) (b)

Figure 3.8: Figures of the (a) original single swipe signal and (b) discrete wavelet transformation reconstructed signal using Daubechies (db1) wavelet.

Then, I have extracted important features from the extracted sub-bands. Specifically, I have extracted entropy values, statistical features, zero crossings and mean crossings from the data readings. Namely, the statistical features are the following, 5th percentile, 25th percentile, 75th percentile, 95th percentile, variance, standard deviation, mean, median and root mean square values of the data. The extracted features vectors for each gesture sample are combined together in NumPy array *data*. Each feature vector contains 168 features. Also, appropriate labels for each feature vector were added in separate NumPy array *classes*.

NumPy is a structure for efficient numerical computation python library described in [51]. NumPy arrays are a representation of numerical data that enable efficient implementation of numerical computations in a high-level language such as Python [51]. Functions that were used to create the features vectors are presented in Listing A.2.

3.2.4 Machine and Deep Learning Gesture Classification

For gestures classification, I used the features described in Section 3.2.3, where the photocurrent signal is analysed in terms of anomalies and variability in signal shapes. Classification refers to the action of providing a machine or deep learning algorithm some features and asking it to assign those features to class, category. In supervised learning, we provide the machine and deep learning classifiers with training data along with labels for that data. In mathematical terms, features are variables that are used to solve a certain equation, problem. Once a model is trained, we use that model on unseen, test, data asking it to predict to which class it belongs. Also, as I have mentioned in Section 3.2.2, the collected files of performed gestures are sorted by gestures name and files have corresponding names of the gesture sample contained within. I have written several functions that automatically extract the class names from the files whilst loading the data of each gesture sample into the Python 3 kernel memory, presented in Listing A.1. The loaded data and classes were then used in 4 supervised machine and

deep learning algorithms from $scikit-learn$ and $keras\ tensorflow$ libraries to learn (classify) the gestures that were defined in Section 3.2.1. Namely, I surveyed the following classifiers, CNN, k-nearest neighbors (KNN) classifier, RF classifier and gradient boosting classifier.

Convolutional Neural Network: Conventional CNNs are usually designed to operate on images that are 2 dimensional (2D) data such as videos and images. CNNs are often referred to as 2D CNNs. However, seeing the remarkable progress of CNNs researches decided to explore its potential on time-series data. For the purpose of time-series classification 1 dimensional (1D) CNNs were developed [52–54]. In the previously mentioned studies, it has been observed that 1D CNNs are superior to their 2D counterparts in cases of 1D signals. This is the case because 1D CNNs perform simple array operations whilst the 2D CNNs perform matrix operations. Moreover, 1D CNNs with simplistic architecture, a small number of hidden layers and neurons, prove to be able to handle well difficult tasks whilst 2D CNNs require much more complex and deeper architecture to achieve similar results. Also, 2D CNNs are much more demanding in terms of computational power in comparison to 1D CNNs. Last but not least important, 1D CNNs provide the ability to implement real-time classification due to the aforementioned advantages which is the goal of this project. Furthermore, in the above-mentioned papers, compact 1D CNNs have demonstrated great performance on time-series signals with limited labelled data and high signal variance, such as patients electrocardiograms (ECGs) and power engines and motors.

A convolution is represented as a sliding filter over the time-series data. And as mentioned above, unlike the cases with 2D CNNs image where we 2D filters (height x width), in 1D CNNs filters have only one dimension, time. In other words, the filters in 1D CNNs are a generic non-linear transformation of the time-series signals.

The architecture of 1D CNNs usually consists of filter layers, activation layers, pooling layers and a final discriminative output layer. The convolutional filters represent the local features of the input time-series signal or feature maps. The activation layer is an activation function that introduces non-linearity in the network and gives it the opportunity to learn more complex models. Popular activation functions are Rectified Linear Unit (ReLU), sigmoid and tanh. Then we have the pooling layer, also called sub-sampling layer because it sub-samples the input feature maps. The main purpose of the pooling layer is to reduce the dimensionality of the feature maps whilst maintaining critical information. Increasing the pooling size will reduce significantly the feature maps however it may lose more valuable data. Lastly, the

output layer takes the representation of the input time-series signal and gives probabilities over the defined class dataset. The output layer is of the size of the number of unique classes in the dataset.

In order to implement a 1D CNN, I have researched numerous papers and websites on time-series classification using deep learning approaches. I have implemented a Sequential model and started with the architecture presented in [53] initially. Their CNN architecture consists of 2 filters (convolution) layers, 2 average pooling layers and 2 fully-connected layers [53]. However, this particular architecture did not perform the best. Then after further research, I have found a great paper that presents a detailed analysis of CNN parameters for time-series signals [52]. Based on their results I improved the architecture of our CNN. The resulting architecture is denoted as 6(7)-3-12(7)-3-64-4 based on the template C1(Size)-S1-C2(Size)-S2-D1-O, where C1 and C2 are numbers of filters in first and second 1D convolution, Size denotes the kernel size, S1 and S2 are subsampling/pooling factors, D1 and O denote the numbers of units in hidden dense and output layers. I have used the ReLU activation function throughout the CNN architecture apart from the last output layer where the softmax function was used because we have 6 classes (gestures) to classify and it is the most suitable for multiclass classification. Also, the average pooling method was used for feature maps sub-sampling after both convolutions. After the 2 convolutions, I have flattened the feature maps and used an additional hidden dense layer in order to increase the number of trainable parameters.

k-Nearest Neighbours Classifier: KNN classifier is an instinctive method where the model classifies new samples, time-series signals in our case, based on their similarities with the training samples. It implements the k-nearest neighbours vote method. In other words, for every new sample X the KNN classifier finds the k nearest neighbourhood in the training samples and labels X as the class that appears most frequently in the k nearest neighbourhood. The KNN classifier provided in the $scikit-learn$ Python library does have many parameters in comparison to a RF classifier for example. It accepts as parameters the size of the neighbourhood *n_neighbors*, the weight function used in prediction *weights*, the algorithm used to compute the nearest neighbours $algorithm$, size of the leaf that is passed to the algorithm used to compute the nearest neighbours $leaf\_size$, the power parameter for the Minkowski metric p, the distance metric to use for the tree $metric$, and other 2 parameters that are not so important *metric_params* and *n_jobs*. For my implementation, I

have started with the default parameters. After multiple experiments, I have found that the neighbourhood of size 17 and the distance weight function performed the best. The distance weight function helped to drastically improve the performance of the classifier on the train data.

Random Forest Classifier: Random forests (RFs) are an ensemble approach for tasks like classification and regression. RFs are constructing multiple decision trees during training and estimate the highest mean probability across the trees for predictions. RFs are an improvement over decision trees as it eliminates the issue with overfitting on the training set by introducing randomness in input and feature selection. As seen from similar projects, such as [13], the RF classifier performs the best at this time for this kind of classification. Also, it is advantageous to use an RF classifier in this case because of the small size of the dataset. The RF classifier used in this project is from the $scikit-learn$ Python library. For the random forest model, I have also started with default parameters and then from conducting multiple training and testing experiments I have found that the following parameters are most suitable for the current dataset. The parameters are, $bootstrap = True$ which makes the random forest model random in terms of how the input data is split and used, $min\_samples\_leaf$ is set to 1 indicating that 1 sample is required to be at a leaf node, 200 $n\_estimators$ also knows as tress in the forest were used first to improve performance, but also the model achieved maximum prediction accuracy at this number, the minimum number of samples required to split an internal node $min\_samples\_split$ was set to 2 as it showed excellent results during testing whilst not constraining the trees to consider more samples at each node, and max_depth depth of each tree in the forest was set 6 to avoid over-fitting of the model, also the model peaked in performance at this value during testing.

Gradient Boosting Classifier: The gradient boosting (GB) classifiers combine a variety of machine learning algorithms that have weak learning models in order to produce a stronger additive model in a forward stage-wise fashion. GB classifiers have the ability of optimisation in relation to different loss functions [55]. Just like the RF classifiers the GB classifiers, are mainly used for classification and regression. Also, similarly to RF classifiers decision trees are usually used in GB classifiers. These classifiers present great effectiveness in the classification of complex datasets. I decided to explore this classifier in addition to the previously mentioned ones in order to determine whether a combination of models would perform better rather a single model which is the case for all of the aforementioned approaches. I used

the GradientBoostingClassifier from the *scikit − learn* Python library. The GB classifier has a great number of parameters that can be tuned that best suit a given dataset. As with the previous models I started with default parameters. After multiple attempts to change multiple parameters, the ones that made a difference and allowed the model to peak in performance were the *n_estimators* and *max_depth* parameters. The *n_estimators* parameter has the same name as the one seen in the RF classifier, but it stands for different proprieties this time. It is indicating the number of boosting stages and was set to 100. The other parameter, *max_depth*, just like in the RF classifier is responsible for the maximum depth of nodes in a tree.

3.3 Demonstrator Application

The proposed system could be emulated as a USB mouse and used in an application with custom toggles, sliders and select drop-down menus. The gestures presented in Figure 3.5 can be mapped to click and scroll actions. In order to test this assumption, I have written a simple HTML page that consists of 3 interactive elements. There are a toggle button, a slider element and a select menu. A screenshot of the application is presented in 3.9.

Figure 3.9: Screenshot of simple HTML page that contains 3 interactive elements.

In order to control the presented interactive elements in 3.9, gestures can be assigned to well-known mouse actions, such as click and scroll. The fist formation gesture depicted in Figure 3.5 (a) can be assigned to perform a click action. Lastly, the 2 circular motion gestures clockwise Figure 3.5 (b) and counterclockwise Figure 3.5 (c) can be assigned to 2 different modes either to mode 1 for mouse movement along *xy* axis, in other words up and down respectively or to mode 2 for scroll up and down actions respectively. The raise lower gesture Figure 3.5 (f) can be used to switch between the above-mentioned 2 modes. In a different scenario, the clockwise and counterclockwise gestures can be used only for mouse movements and the swipe Figure 3.5 (d) and swipes Figure 3.5 (e) gestures for scrolling. The swipe gesture can be used for scrolling down and the swipes gesture for scrolling up.

Chapter 4

Evaluation

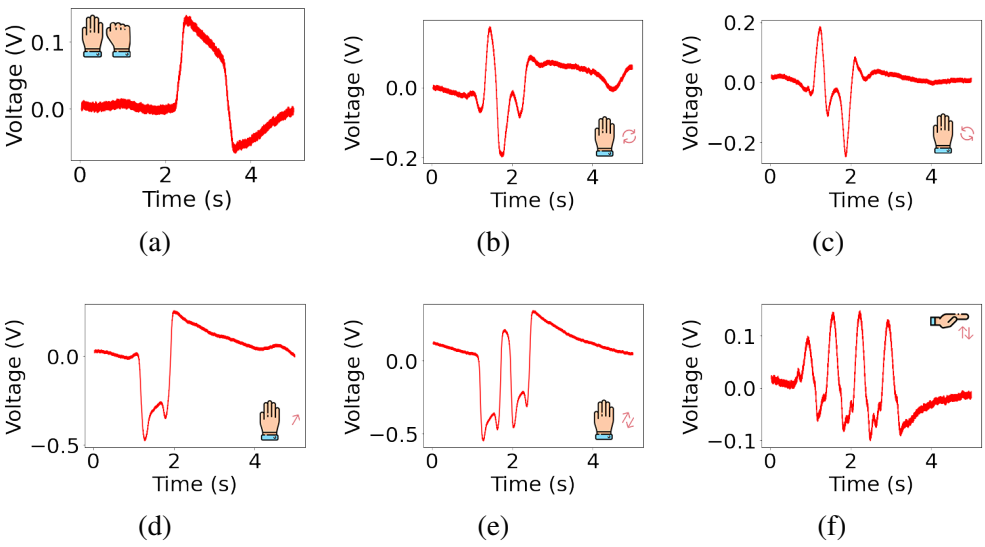


Figure 4.1: Readings of the photocurrent, (a) fist (open/close), (b) Clockwise motion, (c) Counterclockwise motion, (d) Singular swipe motion, (e) Continuous swipes motion, and (f) raise/lower hand motion.

We tested the prototype amongst the authors to create a set of six hand gestures in order to evaluate the prototype. We considered a set of potential use cases such as the one presented in Section 1.4. The defined set of gestures is shown in Figure 3.5. They are (a) fist open/close, circular swipe (b) clockwise and (c) counterclockwise, linear swipe (d) once and (e) continuously and (f) move the hand up/down above the PV sheet as shown in Figure 3.5(f). The gestures were performed in a laboratory environment under standard workplace lighting condition, i.e. with multiple fluorescent lights in the ceiling giving a light power of about 800 lux.

We performed the gestures over five seconds with a natural speed of movement of the hand.

The recognition signals corresponding to the six gestures in Figure 3.5 are shown in Figure 4.1. The signatures corresponding to each gesture are unique. The fist open/close gesture in Figure 4.1 (a) changes the DC level in the signal due to change in the size of the shadow from the hand. The signal shows a spike in the photocurrent which is maintained whilst the fist is not open. Then we observe the voltage dropping as we open the fist. The signal corresponding to the circular swipe gestures in Figure 4.1 (b) and (c) for the clockwise and counterclockwise motions are inverted in the time when compared with each other. The dynamic shadow pattern on the PV sheet involves the increasing/decreasing shadow from the arm. The signal shows a notch on the rising or falling edge. The signals corresponding to the linear swipe (d) once and (e) continuously are also unique. The signal for the continuous swipe gesture consists of multiple padded sequences of the signal corresponding to the signal swipe gesture. These signals have two sharp troughs without notches on the rising or falling edges. The signal corresponding to the (f) move hand up/down gesture changes due to the change in the intensity of shadow from the hand. It consists of rounded peaks and troughs with notches.

These unique signatures were consistently observed for different lighting conditions and position and orientation of the user and the PV sheet. The signals could be calibrated to different light conditions and users as explained in Section 3.2.3 (calibration). For feature extraction, we considered machine learning approaches, such as [56], CNN classifiers that extract features during training, and mathematical functions. The *tsfresh* Python library from [56] was extremely slow due to the scale of the data points in one sample, 15004. It would take hours. A small and simple single-board microcontroller would not be able to handle this library. More suitable algorithms were selected based on complexity, execution time and energy cost as described in Section 3.2.3 (features extraction).

4.1 Classifiers Performance Evaluation

Engineer Train and Test Datasets: Based on the descriptive analysis of machine and deep learning classifiers and their parameters/architecture provided in Section 3.2.4 4 machine and deep learning classifiers have been implemented. Specifically, a random forest (RF) classifier, a k-nearest neighbours (KNN) classifier, a gradient boosting (GB) classifier and a convolutional neural network (CNN) classifier. Sample signals collected and described in Section 3.2.2 have been used for training and testing. Then, data samples of the photocurrent signals were loaded into Python 3 kernel memory and classes were assigned respectively using the methodology

described in Section 3.2.3. Next, the data samples and respective classes were split randomly into train and test sets using the $train\_test\_split$ function from the $scikit-learn$ Python library. From 154 sample files loaded in as Numpy arrays, 23%, 35 random samples, were assigned to the test dataset and 77%, 118 random samples, were added to the train dataset. For the $train\_test\_split$ 42 was used as the random state to separate the dataset.

Features: For the RF, GB and KNN classifiers I have extracted significant features as described in Section 3.2.3 (feature extraction). Features were extracted for train and test datasets separately. The extracted and classified feature vectors were stored into new Numpy arrays that are later used for training and testing. The purpose of feature engineering is to improve the performance of the above-stated classifiers. Crafting features manually will minimise the required computational power as the models are less complex and improve prediction performance. There is a great variety of unique datasets for different purposes. Machine and deep learning classifiers cannot perform the best for all of them. This is why the person who provides the dataset, having better a knowledge of the field from which the specific dataset was collected, can engineer much more meaningful features. Perhaps, in some cases, not all features are of interest.

Training: To train the constructed CNN model, I have used the training dataset without any prior feature extraction. Feature extraction is part of the CNN model training. It automatically extracts various features and throughout convolutions, some are dropped and some are kept for further analysis. A final set of features is presented in the flattening layer. The CNN model was trained using 50 epochs. Increasing the number of epochs above 50 did not improve the results of the model. For the other three classifiers, RF, GB and KNN, features had to be crafted manually as described above in **Features**. The crafted feature vectors were then used to train the defined models. The classifiers were used in a supervised manner to classify the train data. All of the examined classifiers achieved *100%* accuracy on the training set. The CNN model was slowest of them all to finish training. CNNs are trained using epochs. The epochs can be manually adjusted accordingly to how the previous number of epochs performed. An epoch is defined as a one-time full processing of the input data.

Testing: Similarly to training, for the CNN model, the raw, so to speak, test dataset was used to predict classes for the contained within samples. The other models were applied on the feature vectors of the test dataset, again to predict the probabilities of the gestures signals contained within being any of the predefined gestures in Figure 3.5. The RF classifier performed the best out of all the examined classifiers. It achieved *97.2%* overall accuracy results on the

test dataset containing the extracted features vectors, Table 4.1. It also achieved *100%* accuracy on the features vectors dataset of the train dataset. Second best were 2 classifiers, KNN and GB. These 2 classifiers achieved *91.6%* overall accuracy on the features vectors of the test dataset, Table 4.1. Also, just like the RF classifier, the KNN and GB classifiers achieved accuracy on the features vectors of the test dataset was *100%* overall. The CNN model came in last, with *75%* accuracy results on the test dataset, Table 4.1. However, it achieved *100%* accuracy on the train dataset just after 15 epochs with minimum loss.

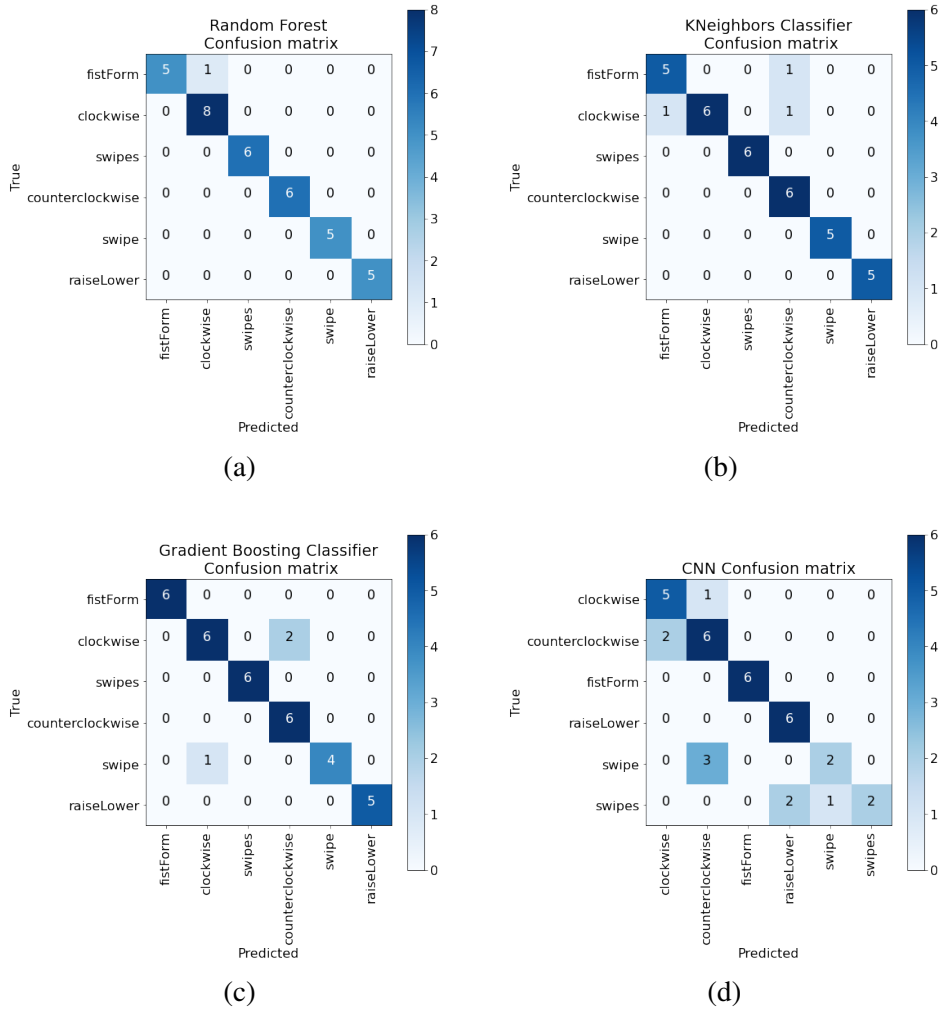
Classifier	**Accuracy**
RF	97.2%
KNN	91.6%
GB	91.6%
CNN	75%

Table 4.1: Presented accuracies of considered classifiers, RF, KNN, GB and CNN, sorted from best to worst.

We see very promising results in terms of accuracies presented in Table 4.1. Each classifier shows great performance in their way. If we look at the confusion matrices of the classifiers in Figure 4.2, we can observe that some models perform great predicting correct classes for some gestures, and not so great for other gestures. This nuance is present in the confusion matrix of each model.

The RF classifier has best results as we can see in Figure 4.2 (a), just one gesture out of 36 was predicted as a wrong gesture when compering to with the true labels. We can see that the fistForm gesture (fist formation, open / close hand) is sometimes mistakenly predicted as the clockwise gesture (clockwise circular hand motion). This hick-up, so to say, can be explained by looking at the data descriptors in Figure 4.3. If we look at the first two tables Figure 4.3 (a) and (b) which represent the descriptions of the fistForm gesture and clockwise gesture, we can observe that the descriptors are not that different within a small range. For example, the mean values of the fistForm gesture data sample are 0.014152 and 0.015031 for the clockwise gesture data sample. On the other hand, the description table of the swipes data sample is very different for all descriptors. Even though there are many similarities between the description tables of fistForm and clockwise gestures, there are also differences which makes the classification possible.

The KNN classifier shows a similar result where the clockwise gesture is mistakenly predicted as the fistForm gesture. With the KNN model, the clockwise and fistForm gestures are also sometimes predicted as the counterclockwise gesture (counterclockwise circular hand motion). This has the same explanation as above. However, it can also be explained by looking at the readings in Figure 4.1. The clockwise gesture Figure 4.1 (b) is the mirrored version of the counterclockwise gesture Figure 4.1 (c) and the fistForm gesture Figure 4.1 (a) has a single peak and sharp drop after the peak which can be observed in the beginning of the counterclockwise gesture Figure 4.1 (c).



(a) (b) (c) (d)

Figure 4.2: Confusion matrices of considered classifiers, (a) Random Forest Classifier confusion matrix, (b) k-Nearest Neighbours Classifier confusion matrix, (c) Gradient Boosting Classifier confusion matrix, (d) Convolutional Neural Network confusion matrix.

	Channel A (V)
count	15004.000000
mean	0.014152
std	0.069606
min	-0.151738
25%	-0.044008
50%	0.013428
75%	0.057558
max	0.206610

(a)

	Channel A (V)
count	15004.000000
mean	0.015031
std	0.079426
min	-0.280587
25%	-0.034425
50%	0.018250
75%	0.069887
max	0.247261

(b)

	Channel A (V)
count	15004.000000
mean	0.035194
std	0.218161
min	-0.627460
25%	-0.007889
50%	0.092776
75%	0.171209
max	0.423109

(c)

Figure 4.3: Example Descriptions of 3 Data Samples, (a) description of data of a fist formation gesture sample, (b) description of data of a clockwise gesture sample, (c) description of data of a swipes gesture sample

The GB classifier presents the same results as the KNN classifier in terms of accuracy percentage. However, we can observe that the fistForm gesture is predicted correctly in all cases. We have again the clockwise gesture mistakenly classified as a counterclockwise gesture. Since the same features vectors were used in both the KNN and GB classifier, the reasoning remains the same as above. Despite the good results for the fistForm gesture predictions, it presents issues in predicting the swipe gesture (singular linear hand motion) correctly. In some cases, the swipe gesture is predicted as a clockwise gesture. This is most likely happening because of the similarity in the gestures shapes after the first peak in the clockwise gesture Figure 4.1 (b). Other than these two cases, the GB classifier showed great performance, it even showed improvements over the previous two classifiers, in predicting fistForm gestures correctly for all test unseen data samples.

Lastly, the worst-performing classifier in terms of prediction accuracy overall was the CNN one. It achieved *75%* overall. While it does not perform the best overall it has achieved better results in individual cases, such as fistForm gesture prediction. It has performed better than the best overall classifier, RF, for this particular gesture. It also demonstrates excellent results for the raiseLower gesture (raise lower hand motion). For the rest of the gestures, the predictions are a bit sporadic. As mentioned previously, this particular model has not performed the best. This can be the case due to several reasons. The architecture of CNN classifier may not be the most suitable for the particular dataset. Also, the dataset itself may need more work in terms

of normalisation and better definition of important features.

4.2 Discussion

Artificial indoor lighting is required for reliable operation of the proposed device. This is particularly suitable in the workplace and overhead lighting environments. Inconsistent lighting in the operating environment poses a limitation. For example, despite artificial indoor lighting, the conditions will vary with the outdoor light conditions during the daytime through the transmitting surfaces like the windows and doors. Therefore, there is more energy that is being harvested with additional outdoor light. Furthermore, the base signal will increase. And since the gesture signals are generated by the shadows from the indoor light, the signal to noise ratio (SNR) will decrease due to the light protruding from outdoors. As a result, the performance of the system might deteriorate during the daytime. The SNR in the current prototype is very low without artificial indoor lights.

Unreliable indoor lighting is also a limiting factor. Different kinds of light sources may introduce unpredictable noise in the gesture recognition signal. By dimming or switching off one or multiple light sources, flickering and light from a temporary external source such as a torch or car headlight could contribute to unreliable behaviour of the proposed system. The proposed device requires adequate light in order to be functional according to the use case scenarios. For example, the device might need low overhead lighting to remain useful during movie watching in a dark room.

The large area of the PV sheet allows the device to be operated in different positions to the ceiling light. Thus, the recognition signal is independent of the position of the shadow on the sheet. Moreover, large gesture input surfaces can be created by connecting multiple PV sheets in series and parallel electrical arrangements. The proposed device is intended to have the ability to recognise the same gestures regardless of the position and orientation of the user, PV sheet and the ceiling lights.

The training to recognise gestures for users can be energy expensive. This is why a decentralized approach was used. The training process is done on a standard computer and models are saved as files that can be imported on low power micro-controllers. However, distributed and centralized computation may lead to trade-offs such as privacy. To address these concerns, a certain level of local computation to access or control the device and its resources is left to be handled by the main board of the system. Also, because the gesture classification is intended to be performed on the micro-controller as shown in Figure 3.3, computationally inexpensive

and low-energy prediction models need to be selected in order to reduce the energy footprint in gesture recognition. The presented approach in Chapter 3 shows promising results and we could also consider going battery free since many functional self-powered devices have been developed, such as battery-less cellphones [46] and cameras [47].

A self-powered device could harvest enough energy for intermittent use. To increase the usage time, low-power energy harvesting micro-controllers are available. In these micro-controllers, different non-essential modules can be put into sleep mode to conserve energy. Combination of a self-powered device and gesture recognition using real-time energy is a challenging task. A possible solution is trickle charging a rechargeable battery or a super-capacitor that would provide enough power on demand. Introducing batteries in the would also pose new challenges. Circuits have to be safe, as batteries pose health danger in case something goes wrong. Also, there should be consideration of how the circuitry can be hidden from the end-user whilst still keeping the system intuitive and easy to set up and use.

Also, despite the promising results of the explored classifiers, the trained classifiers present limitations. Some did not perform as well as expected. The other classifiers that achieved great performance may have troubles when they are presented with partially captured data. For instance, half of the swipes gesture is the same as the swipe gesture. The system can misbehave and start recognising wrong gestures or none at all. In order to mitigate this issue, it is necessary that during deployment we train the system to reject partially captured data, by identifying where the signal starts and ends. Furthermore, the classifiers that did not perform very good can be further improved with more training and by adjusting existing and new parameters. Perhaps, different classifiers could be considered as well. Furthermore, different feature vectors can be considered in order to eliminate the confusion between some of the gestures.

The RF classifier has achieved such excellent performance due to the fact that it does not need as many samples for training as a CNN classifier for instance. However, we cannot discard the 1D CNN model just yet. The data that is fed into the CNN model for training and testing could be prepossessed further. Recent work [57] shows the possibility of transforming the signals into scalograms using continuous wavelet transformations and then classifying it with CNNs. Another way to improve the 1D CNN model is to provide more training data. It will be able to identify more distinct features that exist in multiple samples. We need more signals samples of performed gestures in order to examine the full potential of all considered models. Perhaps even the models that show great performance now will also benefit from a bigger dataset of samples of performed gestures. Samples of performed gestures need to be

collected from as many users as possible to make the system more robust. The explored models can also greatly benefit from the introduction of more variability in the data, such as different light conditions.

Unfortunately, some of the planned work could not be carried out due to the current critical situation around the world. Since the university facilities are not accessible anymore, I lost access to the system and all work was postponed. However, the proposed approach could give a low-cost sensor and computation suitable for decentralized control of IoT devices. The sensor can be integrated into everyday objects such as coffee tables to provide seamless interactive surfaces. The system is still due to be deployed and tested. More work needs to be done in order to achieve satisfactory results that could possibly be published to a journal.

Currently, the achieved results set grounds for future work. There is still room for improvement in terms of the considered classifiers and gestures dataset. However, our current work shows that the potential of using large-area PV sheets as interactive surfaces with indoor lighting could be feasible. The initial proposal of this project has been submitted to and accepted for the workshop on self-powered sustainable interfaces and interactions (SelfSustainableCHI 2020) indicating that the presented topic is of interest in the self-powered sustainable interfaces research community. The performed experiments and the developed code are available on GitHub, https://github.com/christian0101/Photovoltaics-Interactive-Tabletop.

4.2.1 Contributions

The main contributions of this work can be summarized as follows:

- **A PV Prototype**

  We present a prototype that can be set up easily by anyone who would like to explore the same topic or even start a different project based on our results.

- **Dataset of Gestures Signals and Developed Algorithms**

  A dataset of defined gestures and the photocurrent signals of performed gestures during the development of this project are provided along with all the experimental data and developed algorithms in a public repository on GitHub.

- **Parameters for Four Machine and Deep Learning Classifiers**

In this work, we present crafted parameters that allow the four considered machine and deep learning classifier, RF, KNN, GB and CNN, to achieve substantial performance on the given dataset.

- **Evaluation of Trained Models**

  We reviewed four machine and deep learning classifiers based on the created dataset of performed gestures signals for gesture classification and presented their performances.

Chapter 5

Future Work

As discussed above and in Section 4.2, current results set the ground for further research and improvements. Future work involves different aspects of the project. A list of possible ways of how the project can be improved is presented below.

- To demonstrate the effective working of the prototype, it would need to be evaluated with a group of users. Also, an improved set of useful gestures could be established in a design workshop.

- Further experiments for gestures recognition are needed in order to determine if the models can be trained globally for all users or on individual user basis. If it is the case that a new classification model is needed for each user individually, it has to be explored how it can be done effectively. Right now the gesture recognition algorithm could be trained within minutes on the current dataset.

- In addition to the evaluation provided in Chapter 4, the gesture recognition performance needs to be evaluated with different height, position and lighting conditions. Furthermore, the gesture recognition pipeline will need to consider the different speed and variability of gestures from the users.

- The prototype could be further developed to work with multiple hands and users under different use case scenarios. These would require further testing and pilot studies.

- For deployment, the power harvesting and gesture recognition electronics needs to be implemented on a single board computer or alike for real-time hand gesture recognition.

- The provided testing application in Section 3.3 with midair toggles, sliders and dial input features can be deployed with a select group of users. Moreover, more complex interactive elements can be added to the application, in order to test the system under real-world conditions. Perhaps, the system could be tested on existing third-party applications that we use on a day to day basis.

- To deploy the proposed system it will be needed to measure the required power to make the system interactive. We propose the use of a LiPo battery that is continuously charged, but it needs to be tested whether this battery can handle the gesture recognition process. Also, for cases when the produced photocurrent is not enough to recharge the battery have to be examined.

- As discussed in Section 4.2, PV cells have limitations. The cells perform best when we have suitable lighting conditions. However, it needs to be investigated whether PV cells can be used for gesture recognition in poorly-lit environments, such as dim cinema room.

- Also discussed in Section 4.2, inconsistent lighting, protruding light from outside sources such as the sun and headlights of bypassing cars can make the gesture recognition process challenging. This interference should be quantified and analysed.

- Data has to be analysed better to improve the gesture recognition process. For instance, there needs to be a mitigation plan describing how to ignore partial gesture readings in order to avoid confusion. This could be achieved by identifying where the gesture starts and ends.

Chapter 6

Conclusions

With the advent of IoT based ubiquitous interfaces, having the smart buildings understand the human body language with a self-sustained and portable device would greatly benefit the users and make human-computer interactions more intuitive and effortless. We could use the indoor light in our houses for self-powered devices using large PV sheets. The energy harvested could be enough to power interfaces with intermittent use around the building. Self-powered, mobile and low-cost interactive surfaces could be made using large PV sheets. We explored a useful set of six gestures with unique signatures and demonstrated with a prototype made using off-the-shelf components.

We collected photocurrent signals in order to define a dataset of gestures samples. The collected signals were stored in CSV files for convenient and easy access. Moreover, the files were sorted into folders that were named according to the gesture samples contained within. For the implementation, the photocurrent signals were loaded into Numpy arrays and labelled using automated functions in Python. The resulting dataset was randomly split into train and test datasets. Machine and deep learning algorithms were used to classify the collected data based on the defined set of gestures. Specifically, we examined three machine learning classifiers and one deep learning classifier. The three machine learning classifiers are a random forest (RF) classifier, a gradient boosting (GB) classifier and a k-nearest neighbours (KNN) classifier from the $scikit-learn$ Python library. The deep learning classifier used in this project is the 1D convolutional neural network (CNN) from $keras\ tensorflow$ Python library.

First, a 1D CNN model was built and trained on the training dataset. Next, the model was tested against unseen samples to predict their classes, to which gesture each sample belongs. The CNN model achieved a maximum 75% overall accuracy. For the machine learning

classifiers, features had to be manually crafted. Feature vectors for each sample were created by using numerous mathematical functions, such as discrete wavelet transform with the first Daubechies (db1) mother wavelet and calculation of entropy values, statistical features, zero crossings and mean crossings. The extracted feature vectors were used to train the RF, GB and KNN classifiers. All the classifiers achieved *100%* accuracy on predicting classes of the samples that they were trained on. During testing the RF classifier showed best overall accuracy results, *97.2%*. The other two classifiers, KNN and GB, performed the same in terms of overall accuracy results, *91.6%*. Even though not all classifiers performed equally well overall, we can observe that some of these classifiers are better for certain specific gestures, such as fistForm gesture.

With this work, we show the feasibility of large-area PV cells as indoor interactive surfaces. I hope this work will inspire makers, designers and researchers to built new innovative use cases and applications based on self-powered devices using large-area PV cells in the future.

Bibliography

[1] Y. Li, T. Li, R. A. Patel, X.-D. Yang, and X. Zhou, "Self-powered gesture recognition with ambient light," in *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '18. New York, NY, USA: ACM, 2018, pp. 595–608. [Online]. Available: http://doi.acm.org/10.1145/3242587.3242635

[2] A. Varshney, A. Soleiman, L. Mottola, and T. Voigt, "Battery-free visible light sensing," in *Proceedings of the 4th ACM Workshop on Visible Light Communication Systems*, ser. VLCS '17. New York, NY, USA: ACM, 2017, pp. 3–8. [Online]. Available: http://doi.acm.org/10.1145/3129881.3129890

[3] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *CVPR 2011*, June 2011, pp. 1297–1304.

[4] J. Lien, N. Gillian, M. Karagozler, P. Amihood, C. Schwesig, E. Olson, H. Raja, and I. Poupyrev, "Soli: Ubiquitous gesture sensing with millimeter wave radar," *ACM Transactions on Graphics*, vol. 35, pp. 1–19, 07 2016.

[5] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[6] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, p. 84–90, May 2017.

[9] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 2016, pp. 265–283.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. VanderPlas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," *CoRR*, vol. abs/1201.0490, 2012.

[11] W. Lu, Z. Tong, and J. Chu, "Dynamic hand gesture recognition with leap motion controller," *IEEE Signal Processing Letters*, vol. 23, no. 9, pp. 1188–1192, Sep. 2016.

[12] V. Kartsch, S. Benatti, M. Mancini, M. Magno, and L. Benini, "Smart wearable wristband for emg based gesture recognition powered by solar energy harvester," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2018, pp. 1–5.

[13] D. Ma, G. Lan, M. Hassan, W. Hu, M. Upama, A. Uddin, and M. Youssef, "Gesture recognition with transparent solar cells: A feasibility study," 10 2018, pp. 79–88.

[14] D. Ma, G. Lan, M. Hassan, W. Hu, M. B. Upama, A. Uddin, and M. Youssef, "Solargest: Ubiquitous and battery-free gesture recognition using solar cells," in *The 25th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '19. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3300061.3300129

[15] M. Khalilbeigi, J. Steimle, J. Riemann, N. Dezfuli, M. Mühlhäuser, and J. D. Hollan, "Objectop: Occlusion awareness of physical objects on interactive tabletops," in *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces*, ser. ITS '13. New York, NY, USA: ACM, 2013, pp. 255–264. [Online]. Available: http://doi.acm.org/10.1145/2512349.2512806

[16] D. Landerer, D. Bahro, H. Röhm, M. Koppitz, A. Mertens, F. Manger, F. Denk, M. Heidinger, T. Windmann, and A. Colsmann, "Solar glasses: A case study on semitransparent

organic solar cells for self-powered, smart, wearable devices," *Energy Technology*, vol. 5, no. 11, pp. 1936–1945, 2017.

[17] K. Yu, S. Rich, S. Lee, K. Fukuda, T. Yokota, and T. Someya, "Organic photovoltaics: Toward self-powered wearable electronics," *Proceedings of the IEEE*, vol. 107, no. 10, pp. 2137–2154, 2019.

[18] Y. K. Meena, K. Seunarine, D. Sahoo, S. Robinson, J. Pearson, A. Pockett, A. Prescott, S. Thomas, K. Lee, and M. Jones, "Pv-tiles: Towards closely-coupled photovoltaic and digital materials for useful, beautiful and sustainable interactive surfaces," vol. 2020, – 2020, pp. –.

[19] Z. Wen, M.-H. Yeh, H. Guo, J. Wang, Y. Zi, W. Xu, J. Deng, L. Zhu, X. Wang, C. Hu *et al.*, "Self-powered textile for wearable electronics by hybridizing fiber-shaped nanogenerators, solar cells, and supercapacitors," *Science advances*, vol. 2, no. 10, p. e1600097, 2016.

[20] T. Grosse-Puppendahl, S. Hodges, N. Chen, J. Helmes, S. Taylor, J. Scott, J. Fromm, and D. Sweeney, "Exploring the design space for energy-harvesting situated displays," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, ser. UIST '16. New York, NY, USA: ACM, 2016, pp. 41–48. [Online]. Available: http://doi.acm.org/10.1145/2984511.2984513

[21] A. Amaravati, S. Xu, N. Cao, J. Romberg, and A. Raychowdhury, "A light-powered smart camera with compressed domain gesture detection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 3077–3085, 2017.

[22] T. Chakraborty, M. Nasim, S. M. B. Malek, M. T. H. Majumder, M. S. Saeef, and A. A. Al Islam, "Devising a novel visible light based low-cost ultra-low-power gesture recognition system," in *2017 International Conference on Networking, Systems and Security (NSysS)*. IEEE, 2017, pp. 3–11.

[23] T. Li, Q. Liu, and X. Zhou, "Practical human sensing in the light," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, 2016, pp. 71–84.

[24] X. Zhou and A. T. Campbell, "Visible light networking and sensing," in *Proceedings of the 1st ACM workshop on Hot topics in wireless*, 2014, pp. 55–60.

[25] P. H. Pathak, X. Feng, P. Hu, and P. Mohapatra, "Visible light communication, networking, and sensing: A survey, potential and challenges," *IEEE communications surveys & tutorials*, vol. 17, no. 4, pp. 2047–2077, 2015.

[26] S. Zhang, K. Liu, Y. Ma, X. Huang, X. Gong, and Y. Zhang, "An accurate geometrical multi-target device-free localization method using light sensors," *IEEE Sensors Journal*, vol. 18, no. 18, pp. 7619–7632, 2018.

[27] Y. Geng, J. Chen, R. Fu, G. Bao, and K. Pahlavan, "Enlighten wearable physiological monitoring systems: On-body rf characteristics based human motion classification using a support vector machine," *IEEE transactions on mobile computing*, vol. 15, no. 3, pp. 656–671, 2015.

[28] J. Wang, Q. Gao, P. Cheng, Y. Yu, K. Xin, and H. Wang, "Lightweight robust device-free localization in wireless networks," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 10, pp. 5681–5689, 2014.

[29] J. Nino-Castaneda, A. Frías-Velázquez, N. B. Bo, M. Slembrouck, J. Guan, G. Debard, B. Vanrumste, T. Tuytelaars, and W. Philips, "Scalable semi-automatic annotation for multi-camera person tracking," *IEEE Transactions on Image Processing*, vol. 25, no. 5, pp. 2259–2274, 2016.

[30] A. De Angelis, A. Moschitta, P. Carbone, M. Calderini, S. Neri, R. Borgna, and M. Peppucci, "Design and characterization of a portable ultrasonic indoor 3-d positioning system," *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 10, pp. 2616–2625, 2015.

[31] S. Hijikata, K. Terabayashi, and K. Umeda, "A simple indoor self-localization system using infrared leds," in *2009 sixth international conference on networked sensing systems (INSS)*. IEEE, 2009, pp. 1–7.

[32] Z. Xu, W. Li, J. Huang, X. Guo, Q. Liu, R. Yu, W. Miao, B. Zhou, W. Guo, and X. Liu, "Flexible, controllable and angle-independent photoelectrochromic display enabled by smart sunlight management," *Nano Energy*, vol. 63, p. 103830, 2019.

[33] X. Cao, A. D. Wilson, R. Balakrishnan, K. Hinckley, and S. E. Hudson, "Shapetouch: Leveraging contact shape on interactive surfaces," in *2008 3rd IEEE International Workshop on Horizontal Interactive Human Computer Systems*. IEEE, 2008, pp. 129–136.

[34] J. Wei, X. Wang, R. L. Peiris, Y. Choi, X. R. Martinez, R. Tache, J. T. K. V. Koh, V. Halupka, and A. D. Cheok, "Codine: an interactive multi-sensory system for remote dining," in *Proceedings of the 13th international conference on Ubiquitous computing*, 2011, pp. 21–30.

[35] F. Echtler and R. Wimmer, "The interactive dining table," *Proc. Blend*, vol. 13, 2013.

[36] ——, "The interactive dining table, or pass the weather widget, please," in *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*, ser. ITS '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 419–422. [Online]. Available: https://doi.org/10.1145/2669485.2669525

[37] J. Rekimoto, "Smartskin: an infrastructure for freehand manipulation on interactive surfaces," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, 2002, pp. 113–120.

[38] D. R. Sahoo, K. Hornbæk, and S. Subramanian, "Tablehop: An actuated fabric display using transparent electrodes," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ser. CHI '16. New York, NY, USA: ACM, 2016, pp. 3767–3780. [Online]. Available: http://doi.acm.org/10.1145/2858036.2858544

[39] H. Manabe and M. Fukumoto, "Touch sensing by partial shadowing of pv module," in *Adjunct Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST Adjunct Proceedings '12. New York, NY, USA: ACM, 2012, pp. 7–8. [Online]. Available: http://doi.acm.org/10.1145/2380296.2380301

[40] T. Li, C. An, Z. Tian, A. T. Campbell, and X. Zhou, "Human sensing using visible light communication," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '15. New York, NY, USA: ACM, 2015, pp. 331–344. [Online]. Available: http://doi.acm.org/10.1145/2789168.2790110

[41] R. H. Venkatnarayan and M. Shahzad, "Gesture recognition using ambient light," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 2, no. 1, pp. 40:1–40:28, Mar. 2018. [Online]. Available: http://doi.acm.org/10.1145/3191772

[42] J. Gong, Y. Zhang, X. Zhou, and X.-D. Yang, "Pyro: Thumb-tip gesture recognition using pyroelectric infrared sensing," in *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, 2017, pp. 553–563.

[43] J. Epps, S. Lichman, and M. Wu, "A study of hand shape use in tabletop gesture interaction," vol. 2006, 04 2006, pp. 748–753.

[44] R. Nacheva, "Principles of user interface design: Important rules that every designer should follow," 10 2015.

[45] G. Knier, "How do photovoltaics work?" 2008, visited on 29-11-2019. [Online]. Available: https://science.nasa.gov/science-news/science-at-nasa/2002/solarcells

[46] V. Talla, B. Kellogg, S. Gollakota, and J. R. Smith, "Battery-free cellphone," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 1, no. 2, pp. 25:1–25:20, Jun. 2017. [Online]. Available: http://doi.acm.org/10.1145/3090090

[47] S. Naderiparizi, A. N. Parks, Z. Kapetanovic, B. Ransford, and J. R. Smith, "Wispcam: A battery-free rfid camera," in *2015 IEEE International Conference on RFID (RFID)*, April 2015, pp. 166–173.

[48] M.-D. Kaholokula, "Reusing ambient light to recognize hand gestures," 2016. [Online]. Available: https://www.cs.dartmouth.edu/~trdata/reports/TR2016-797.pdf

[49] G. Lee, R. Gommers, F. Waselewski, K. Wohlfahrt, and Aaron, "Pywavelets: A python package for wavelet analysis," *Journal of Open Source Software*, vol. 4, p. 1237, 04 2019.

[50] D. Li, T. F. Bissyandé, J. Klein, and Y. L. Traon, "Time series classification with discrete wavelet transformed data: Insights from an empirical study," in *SEKE*, 2016.

[51] S. van der Walt, S. Colbert, and G. Varoquaux, "The numpy array: A structure for efficient numerical computation," *Computing in Science & Engineering*, vol. 13, pp. 22 – 30, 05 2011.

[52] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu, "Convolutional neural networks for time series classification," *Journal of Systems Engineering and Electronics*, vol. 28, no. 1, pp. 162–169, 2017.

[53] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Time series classification using multi-channels deep convolutional neural networks," in *Web-Age Information Management*, F. Li, G. Li, S.-w. Hwang, B. Yao, and Z. Zhang, Eds. Cham: Springer International Publishing, 2014, pp. 298–310.

[54] ——, "Exploiting multi-channels deep convolutional neural networks for multivariate time series classification," *Frontiers of Computer Science*, vol. 10, no. 1, pp. 96–112, 2016.

[55] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers in neurorobotics*, vol. 7, p. 21, 12 2013.

[56] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, "Time series feature extraction on basis of scalable hypothesis tests (tsfresh – a python package)," *Neurocomputing*, vol. 307, pp. 72 – 77, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231218304843

[57] A. Meintjes, A. Lowe, and M. Legget, "Fundamental heart sound classification using the continuous wavelet transform and convolutional neural networks," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2018, pp. 409–412.

Appendix A

Implementation of Various Functions

```
'''
Extract filename from a given path.

Input:
    _path - given path from which name has to be extracted

Output:
    extracted file name

'''
def path_to_fname(_path):
    return os.path.basename(os.path.splitext(_path)[0])

'''
Remove digits and underscore _ from a given string.

Input:
    _string - string to be processed

Output:
    resulting string without digits and _

'''
def rm_digit_undscore(_string):
    return ''.join(c for c in _string if not c.isdigit() and not c == '_')


'''
Load data from a given file list.

Input:
```

```
    file_list - list of files

Output:
    classes - class names for each loaded file
    gestures - gestures photocurrent (V) values

'''
def load_data(file_list):
    classes = []
    gestures = []

    for fp in file_list:
        gesture = pd.read_csv(fp, delim_whitespace=False, header=[0, 1], index_col
            =0)
        gesture = gesture.rolling(window = 100).mean()
        gesture = gesture.dropna()
        gestures.append(gesture.values)
        classes.append(rm_digit_undscore(path_to_fname(fp)))

    gestures = np.array(gestures)
    classes = np.array(classes)
    return classes, gestures
```

Listing A.1: Implementation of automated data and class extraction from sample files.

```
'''
    Below are function extracting features which are most frequently used for time
         series signals.

    Auto-regressive model coefficient values

    Entropy values - entropy values can be taken as a measure of complexity of the
         signal

    Statistical features like:
        variance
        standard deviation
        Mean
        Median
        25th percentile value
        75th percentile value
        Root Mean Square value;
        square of the average of the squared amplitude values
        The mean of the derivative
        Zero crossing rate, i.e. the number of times a signal crosses y = 0
        Mean crossing rate, i.e. the number of times a signal crosses y = mean(y)
'''
```

```
# source: http://ataspinar.com/2018/12/21/a-guide-for-using-the-wavelet-transform-in-machine-learning/
def calculate_entropy(list_values):
    counter_values = Counter(list_values).most_common()
    probabilities = [elem[1] / len(list_values) for elem in counter_values]
    entropy = scipy.stats.entropy(probabilities)
    return entropy

# source: http://ataspinar.com/2018/12/21/a-guide-for-using-the-wavelet-transform-in-machine-learning/
def calculate_statistics(list_values):
    n5 = np.nanpercentile(list_values, 5)
    n25 = np.nanpercentile(list_values, 25)
    n75 = np.nanpercentile(list_values, 75)
    n95 = np.nanpercentile(list_values, 95)
    median = np.nanpercentile(list_values, 50)
    mean = np.nanmean(list_values)
    std = np.nanstd(list_values)
    var = np.nanvar(list_values)
    rms = np.nanmean(np.sqrt(list_values**2))
    return [n5, n25, n75, n95, median, mean, std, var, rms]

# source: http://ataspinar.com/2018/12/21/a-guide-for-using-the-wavelet-transform-in-machine-learning/
def calculate_crossings(list_values):
    zero_crossing_indices = np.nonzero(np.diff(np.array(list_values) > 0))[0]
    no_zero_crossings = len(zero_crossing_indices)
    mean_crossing_indices = np.nonzero(np.diff(np.array(list_values) > np.nanmean(
        list_values)))[0]
    no_mean_crossings = len(mean_crossing_indices)
    return [no_zero_crossings, no_mean_crossings]

# source: http://ataspinar.com/2018/12/21/a-guide-for-using-the-wavelet-transform-in-machine-learning/
def get_features(list_values):
    entropy = calculate_entropy(list_values)
    crossings = calculate_crossings(list_values)
    statistics = calculate_statistics(list_values)
    return [entropy] + crossings + statistics

'''
Extract features for each gesture in the given dataset.

Input:
    dataset - dataset containing the gestures signals
    labels - gestures labels for elements in the given 'dataset'
    waveletname - name of wavelet for dwt
```

```
63
64  Output:
65      data - extracted features vectors
66      classes - classes identifying each feature vector in 'data'
67
68  '''
69  def get_gestures_features(dataset, labels, waveletname):
70      all_features = []
71      dataset = dataset.reshape(dataset.shape[0], dataset.shape[2])
72      for i in range(len(dataset)):
73          features = []
74          data_i = dataset[i, :]
75          list_coeff = pywt.wavedec(data_i, waveletname)
76          for coeff in list_coeff:
77              features += get_features(coeff)
78          all_features.append(features)
79
80      data = np.array(all_features)
81      data = np.expand_dims(data, axis=[1,-1])
82      classes = np.array(labels)
83
84      return data, classes
```

Listing A.2: Feature Extraction Functions for Machine Learning Classifiers.