CHAPTER 1

Methods

With a minimum viable prototype established, it now becomes possible to devise experimental scenarios to test against the project's goals. The project's adherence to the goals of Low-Cost and Non-Invasiveness have been evaluated previously, so in this section we will focus on the project's adherence to Reliability and Energy Efficiency goals.

1.1 Reliability Testing

With the prototype, it is now possible to utilize the prototype to gather both thermal and visual data in a synchronized format. This data can be collected and used to determine the effectiveness of the human counting algorithms used. Due to the prototype's technical similarly to Thermosense [1], a similar set of experimental conditions will be used, with a comparison against Thermosense being used as a benchmark. To this end, several experiments were devised, each of which had its data gathered and processed in accordance with the same general process, outlined in Figure 1.1 on the preceding page.

1.1.1 Data gathering

As the camera and the Arduino are directly plugged into the Raspberry Pi, all data capture is performed on-board through SSH, with the data being then copied of the Pi for later processing. To perform this capture, the main script used is `capture_pi_synced.py`.

`capture_pi_synced.py` takes two parameters on the command line; the name of the capture output, and the number of seconds to capture. By default, it always captures at 2Hz. The script initializes the `picamera` library, then passes a reference to it to the `capture_synced` function within the `Visualizer` class.

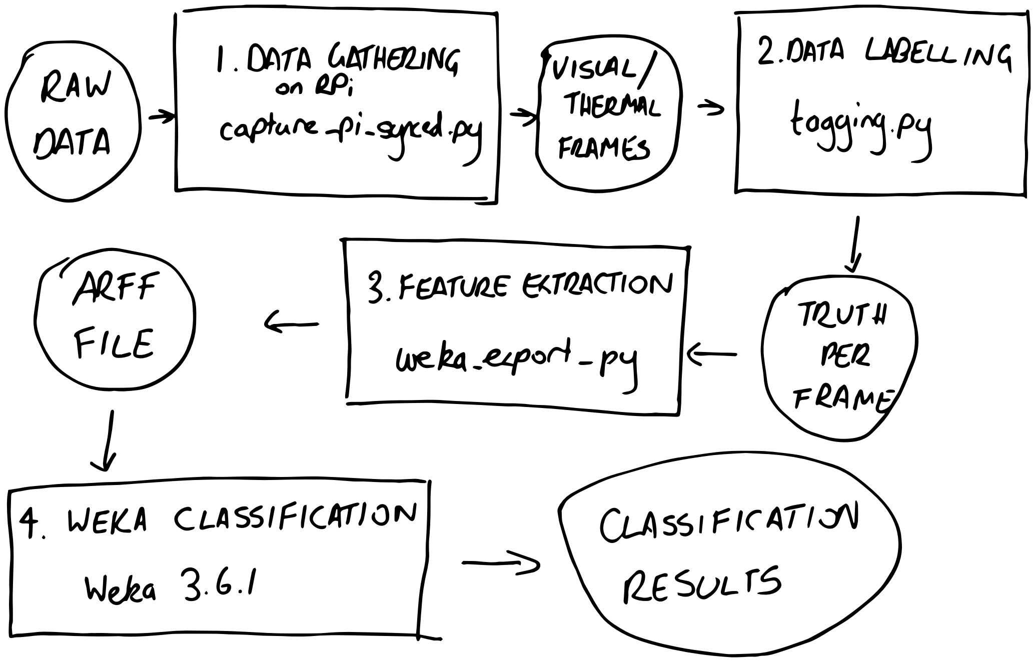

Figure 1.1: Flowchart of processing

The class will then handle the sending of commands to the Arduino to capture data in concert with taking still frames with the Raspberry Pi's camera.

When the script runs, it creates a folder with the name specified, storing inside a file named `output_thermal.hcap` containing the thermal capture, and a sequence of files with the format `video-%09d.jpg`, corresponding to each visual capture frame.

1.1.2 Data labeling

Once this data capture is complete, the data is copied to a more powerful computer for labeling. The utility `tagging.py` is used for this stage. This script is passed the path to the capture directory, and the number of frames at the beginning of the capture that are guaranteed to contain no motion. This utility will display frame by frame each visual and thermal capture together, as well as the computed feature vectors (based on a background map created from the first n frames without motion).

The user is then required to press one of the number keys on their keyboard to indicate the number of people present in this frame. This number will be recorded in a file called `truth` in the capture directory. The next frame will then be displayed, and the process continues. This utility enables the quick input of the ground truth of each capture, making the process more efficient.

1.1.3 Feature extraction and data conversion

Once the ground truth data is available, it is now possible to utilize the data to perform various classification tests. For this, we use version 3.6.12 of the open-source Weka toolkit [2], which provides easy access to a variety of machine learning algorithms and the tools necessary to analyze their effectiveness.

To enable the use of Weka, we export the ground truth and extracted features to Weka Attribute-Relation File Format (ARFF) for processing. `weka_export.py` takes two parameters, a comma-separated list of different experiment directories to pull ground truth and feature data from, and the number of frames at the beginning of each capture that can be considered as "motionless." With this information, a CSV-file file is generated on which the heading from Listing **??** on page ?? is added for Weka to recognize.

```
@RELATION persondata

@ATTRIBUTE npeople  {0,1,2,3} % Or attribute could be numeric using
 ↪  "NUMERIC"
@ATTRIBUTE numactive   NUMERIC
@ATTRIBUTE numconnected  NUMERIC
@ATTRIBUTE sizeconnected   NUMERIC

@DATA
```

Listing 1.1: ARFF Header

1.1.4 Running Weka Tests

Once the ARFF file is generated, it is then possible to open the file in Weka for processing. Weka provides a variety of algorithms, but we choose a specific subset of algorithms based on those present in the Thermosense paper [1], as well others that we believe adequately represent the different approaches to classification.

We perform the following Weka classification tests on the dataset;

Type	**Attribute**	**Weka Class & Parameters**
Neural Net	Nominal, Numeric	`weka.classifiers.functions.MultilayerPerceptron` `-L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a`

K-Neareast Neighbours	Nominal, Numeric	`weka.classifiers.lazy.IBk`
		`-K 1 -W 0`
		`-A "weka.core.neighboursearch.LinearNNSearch -A`
		`\"weka.core.EuclideanDistance -R first-last\""`

Naive Bayes	Nominal	`weka.classifiers.bayes.NaiveBayes`

Support Vector Machine	Nominal	`weka.classifiers.functions.SMO`
		`-C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1`
		`-K "weka.classifiers.functions.supportVector.PolyKernel`
		`-C 250007 -E 1.0"`

Decision Tree	Nominal	`weka.classifiers.trees.J48` `-C 0.25 -M 2`
Entropy Distance	Nominal	`weka.classifiers.lazy.KStar` `-B 20 -M a`
Linear Regression	Numeric	`weka.classifiers.functions.LinearRegression` `-S 0 -R 1.0E-8`
Decision Stump	Numeric	`weka.classifiers.trees.DecisionStump`

Table 1.1: Weka classifiers used with parameters

For those tests that are "nominal," the `npeople` attribute was set to $\{$`0,1,...,n`$\}$ where n is the maximum number of people detected in the classification data. For those tests that are "numeric," `npeople` was set to `NUMERIC`. For all tests, we use 10-fold cross-validation to validate our results.

As the data we are using is based on real experiments, the number of frames which are classified as each class may be unbalanced, which could cause the classification results to be affected. To that end, for each classification technique, we both classify the data in its raw, unbalanced form, and we also uniformly resample the `npeople` parameter using `weka.filters.supervised.instance.Resample -B 1.0 -S 1 -Z 100.0` in the pre-processing stage.

1.1.5 Classifier Experiment Set 1 Setup

In our first set of experiments, a scene was devised in accordance with Figure 1.2 on page 7 that attempted to sense people from above, as did Thermosense. The prototype was set up on the ceiling, pointing down at a slight angle. For ease of use, the prototype was powered by mains power, and was networked with a laptop for command input and data collection via Ethernet. This set of experiments involved between zero and three people being present in the scene, moving in and out in various ways in accordance with the script in Table **??**.

1. (Remained standing) One person walks in, stands in center, walks out of frame.

2. (Remained standing) One person walks in, joined by another person, both stand there, one leaves, then another leaves.

3. (Remained standing) One person walks in, joined by one, joined by another, all stand there, one leaves, then another, then another.

4. (Remained standing) Two people walk in simultaneously, both stand there, both leave simultaneously.

5. (Sitting) One person walks in, sits in center, moves to right, walks out of frame.

6. (Sitting) One person walks in, joined by another person, both sit there, they stand and switch chairs, one leaves, then another leaves.

7. (Sitting) One person walks in, joined by one, joined by another, they all sit there, one leaves, then another, then another.

8. (Sitting) Two people walk in, both sit there, both leave.

Table 1.2: Experiment Set 1 Script

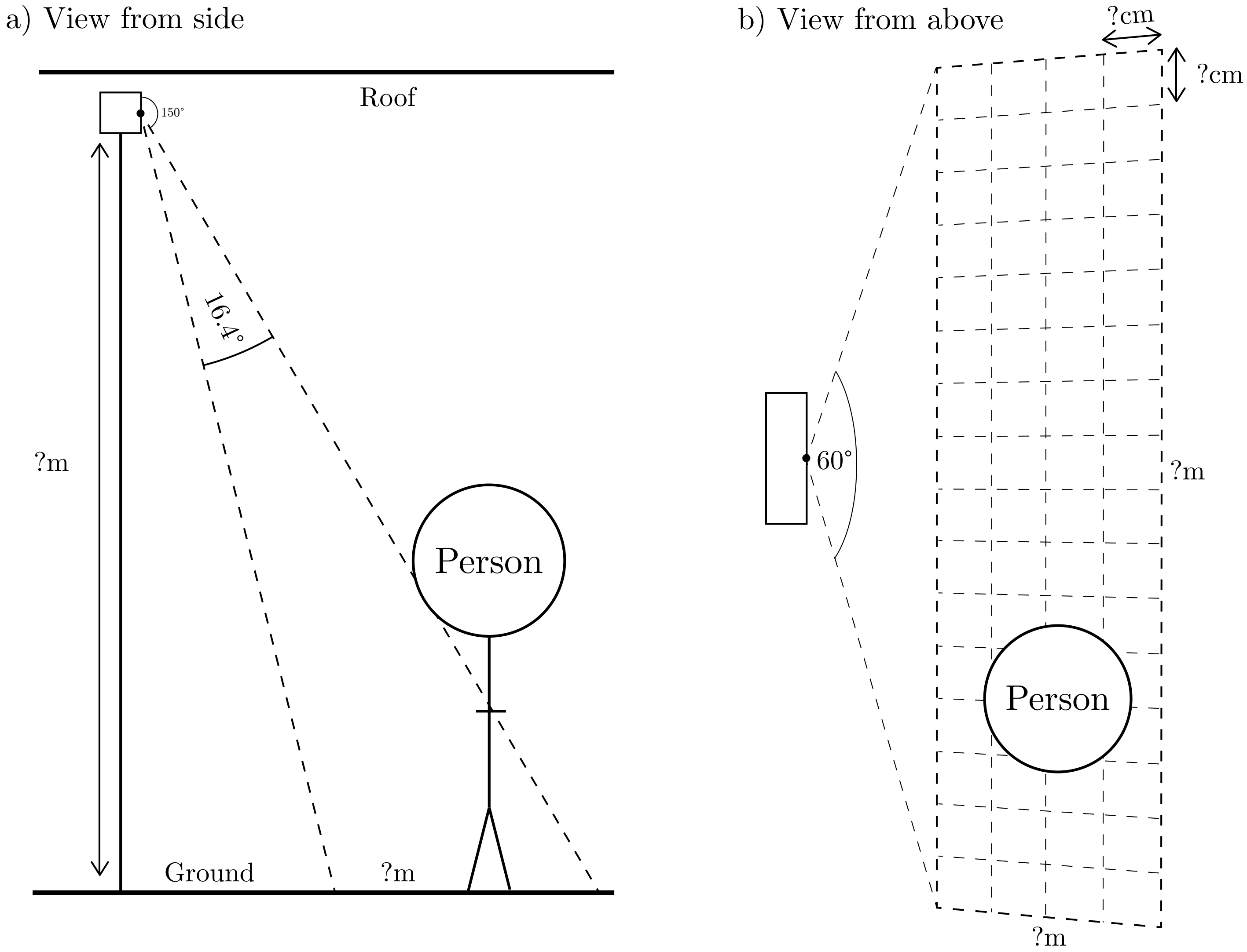

Figure 1.2: Classifier Experiment Set 1 Setup

In these experiments people moved slowly and deliberately, making sure there were large pauses between changes of action. The people involved were of average height, wearing various clothing. The room was cooled to 18 degrees for these experiments.

Each experiment was recorded with a thermal-visual synchronization at 1Hz over approximately 60-120 second intervals. Each experiment had 10-15 frames at the beginning where nothing was within the view of the sensor to allow the thermal background to be calculated. Each frame generated from these experiments was manually tagged with the ground truth value of its occupancy using the script mentioned previously.

The resulting features and ground truth were combined and exported to ARFF allowing the Weka machine learning program to analyze them. This data was analyzed with the feature vectors always being considered numeric data and with the ground truth considered both numeric and nominal (nominal being `0,1,2,3`). All previously mentioned classification algorithms were run against the data set.

1.2 Energy Efficiency Testing

Bibliography

[1] Beltran, A., Erickson, V. L., and Cerpa, A. E. ThermoSense: Occupancy thermal based sensing for HVAC control. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings* (2013), ACM, pp. 1–8.

[2] University of Waikato. Weka. `http://www.cs.waikato.ac.nz/ml/weka/`. Accessed: 2015-03-10.