Recognition and Tracking Analytics for Crowd Control

September 2, 2016

Abbad Vakil, Saransh Kacharia, Brian Liao, Ibrahim Mohiuddin, Haytham Shaban

Foundry10

Abstract

We explore and apply methods of image analyzation in several forms in order to monitor the condition and health of a crowd. Stampedes, congestion, and traffic all occur as a result of inefficient crowd management. Our software identifies congested areas and determines solutions to avoid congestion based on live data. The data is then processed by a local device which is fed via camera. This method was tested in simulation and proved to create a more efficient and congestion-free scenario. Future plans include depth sensing for automatic calibration and suggested course of action.

1 Introduction

Throughout this paper, we will be describing in depth the crowd analytics software and hardware applications of Project Icarus, a method to identify congestion and crowding in open areas. We targeted at crowd control teams, event planners, and product developers incorporating the software to various hardware devices and scenarios.

- An overview of Icarus features
- A systematic overview of the Icarus software, including processing methods
- The analyzation process and methods used to collect the data input from camera sources
- The application of live results/suggestions

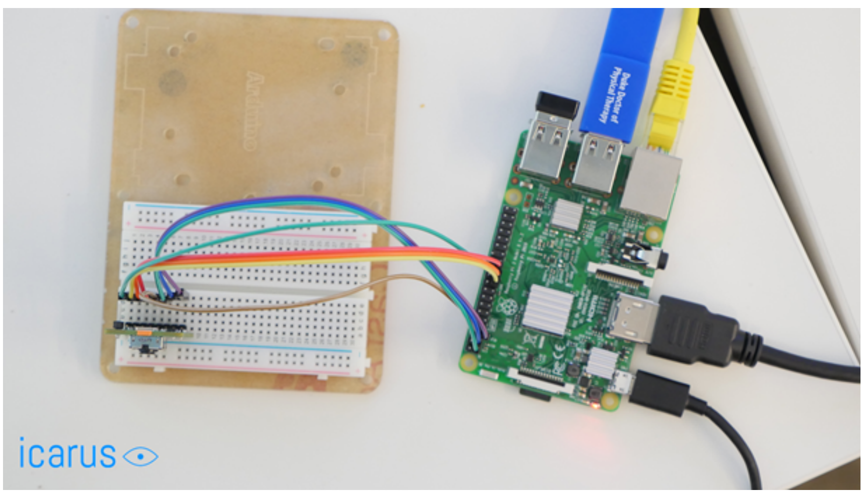

Fig. 1. Raspberry Pi and the Lepton Flir thermal camera

1.1 Background

In the twenty-first century alone, over sixty-one incidents of human stampedes have occurred across the globe. A commonly overlooked problem, crowd congestion, is an instantaneous occurrence that can have deadly consequences, if not managed preemptively. Of these incidents, five have occurred in Saudi Arabia, primarily during the time of rituals taking place in Mecca. Mecca's most recent incident was the 2015 Mina human crush. Due to inefficient crowd management, over two million pilgrims were escorted to the same location at the

same time of day. A human crush was bound to occur, and over 2,177 pilgrims lost their lives that day.

2 What is Project Icarus?

Icarus is a crowd analytics software designed to pull live data from a venue/crowd and apply the data in a meaningful way. We utilizes a thermal camera for live frame captures, then processes the images into data that crowd control teams and event planners can use to manage crowds more efficiently. We are able to recognize where crowds of people are located, the size of the crowd, the speed of the crowd, and tracking of the crowd over time.

Using the "FLIR Lepton" thermal camera, frames are captured and analyzed using project Icarus' blob detection software. When the program is executed, the "whiter" parts of the image are detected and ID'd with different colors. This is used to find the size of the crowds and their midpoints, which are found by averaging the locations of each of the pixels in a given blob.

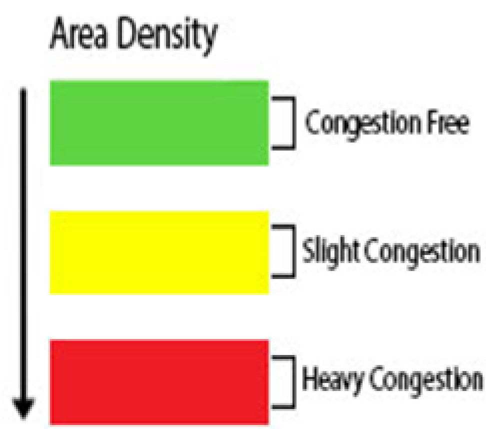

Fig. 2. Levels of congestion.

2.0.1 Density Key

When a "blob" is recognized, and the proximity is known, we are able to calculate how much area is occupied per person. The further a person is from the standard of one square meter, the more saturated its area is in the color of red. This model provides information for efficient and simplistic identification.

For high accuracy, samples are collected one after another and analyzed repeatedly. The feed received from the camera is processed on the hardware device and the results are returned to a server.

2.1 Feed Information

Along with receiving data for density from live feed represented by the coloration, the data collected can be used for several applications.

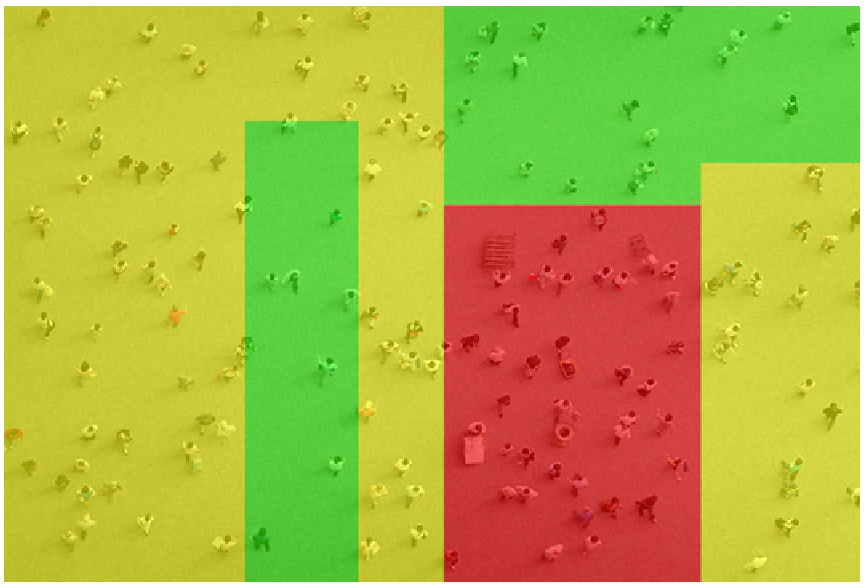

Fig. 3. DIfferent congestions in a crowd.

- The processed image shows:
- Congested or highly dense area into sub squares of the larger image
- The **green** represents are as with close to no congestion
- The **yellow** represents areas with low congestion, potentially a place where the program will not funnel the crowd
- The **red** represents an area which needs to be managed

2.2 Thermal Camera Identification

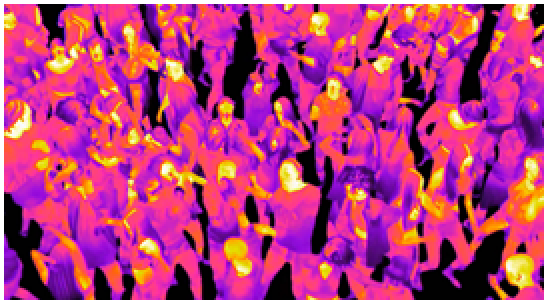

Fig. 4. Sample thermal image.

Using Python's OpenCV, we are able to take pictures one after another, and process them consecutively.

```
from pylepton import Lepton

with Lepton() as l:
        a,_ = l.capture()
```

This model allows for tracking moving people along with people at stand still. Using the thermal feed in conjunction with the analyzation software allows for thorough results. The thermal feed also monitors a video feed showing immediate changes in a group of people. This model is accurate for long range cameras as well as short range ones.

Fig. 5. Different blobs in a thermal image.

A recursive algorithm is applied to each pixel that is hot enough to be considered a blob.

```
def analyze():
    global upCount, rightCount, leftCount, downCount, xMidSum, yMidSum, xSum, ySum,
currentSize, r, g, b, image
    importantIndexes = []
    blobCount = 0
    tempBlobs = blobs
    for y in xrange(image.size[1]):
        for x in xrange(image.size[0]):
            if (pixdata[x, y][0]+pixdata[x, y][1]+pixdata[x, y][2]>600):
                pixdata[x,y] = (0,255,0)
                importantIndexes.append([x, y])
    blobCount = 0
    for pixel in importantIndexes:
        if (checkValid(pixel[0],pixel[1])):
            reset()
            r = randint(0, 254)
            g = randint(0, 254)
            b = randint(0, 254)
            pixdata[pixel[0],pixel[1]] = (r,g,b)
            blobCount+=1
            fillAll(pixel[0],pixel[1])
            if(xSum == 0):
```

3 Tracking Blobs

Once blobs are created, they are ID'd and tracked frame by frame. A midpoint is assigned and is used to track the blob over time. Once the next image is taken, a function searches in the same location as the previous midpoint and if it is in a certain radius of the previous point, it is assigned the same identity and color. A line is also drawn using HTML5 Canvas which shows the movement of the blob and its previous locations.

Tracking and midpoints turned on.

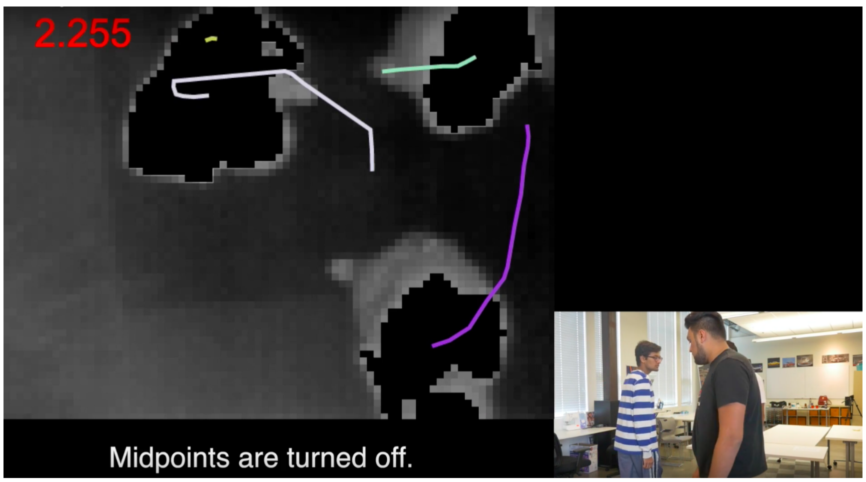

Fig. 6: Midpoints turned off.

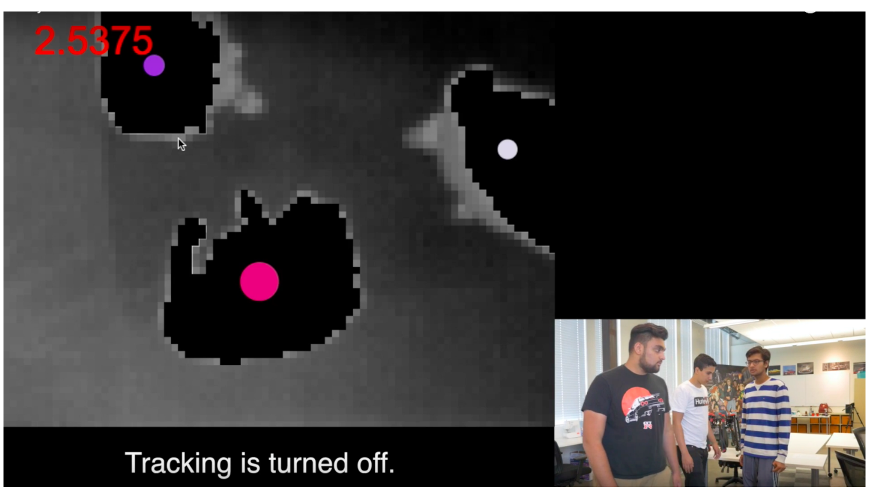

Fig 7: Tracking turned off but midpoints turned on.

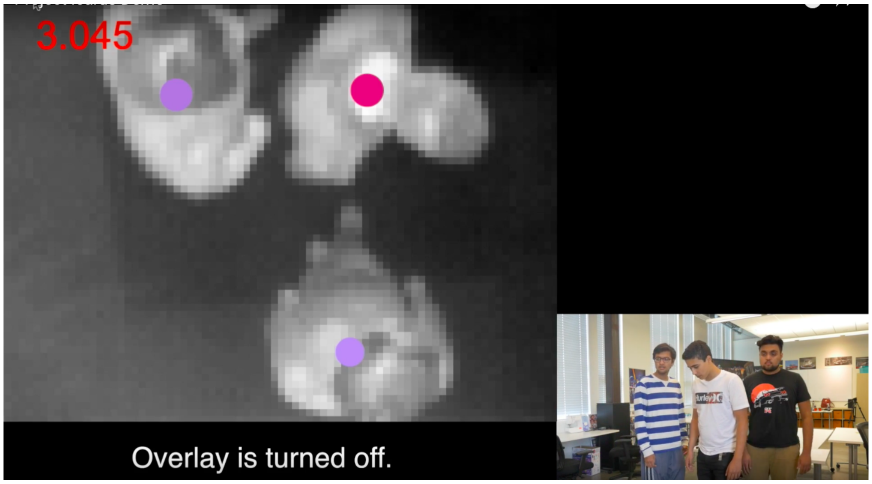

Fig 8: Overlay turned off.

4 Icarus Feature Set

Using a “FLIR Lepton” thermal camera, instances of data retrieved from footage at a capture rate specified in the program. When the program is executed, the algorithm scans all heat signatures, incrementing by how much mass a certain threshold of thermal readings are covering, and calculating the distance from the crowd to provide an accurate number of people in that area. The algorithm then individualizes each group and attaches a form of identification.

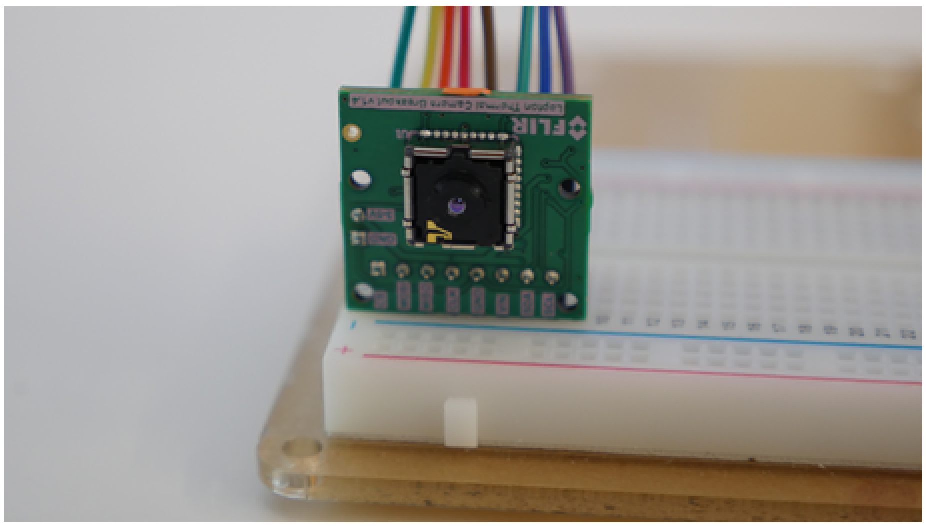

Fig. 9. Flir Lepton thermal camera.

The main features of Icarus are as follows:

- Scans area for heat signatures
- Groups areas of heat based on how much area of the same threshold is connected (“various shapes of different sizes will appear based on the formation of the crowd)
- Recognizes each shape as a separate unit
- Locates congestion points by scanning movement of all shapes and finding areas moving at a slower pace than expected
- Sends information to “raspberry pi” motherboard, accesses database for layout of intended crowd formation, and map of the area if stored
- Compares movement of crowd to expected route, and delivers visual/audial signals for the crowd to follow accordingly

5 Project Icarus System Overview

Project Icarus has four main components aside from the software side. The thermal camera captures a feed which is processed by the Raspberry Pi. After processing, the data is sent to a server which packages the results in a JSon query to be used for the client end.

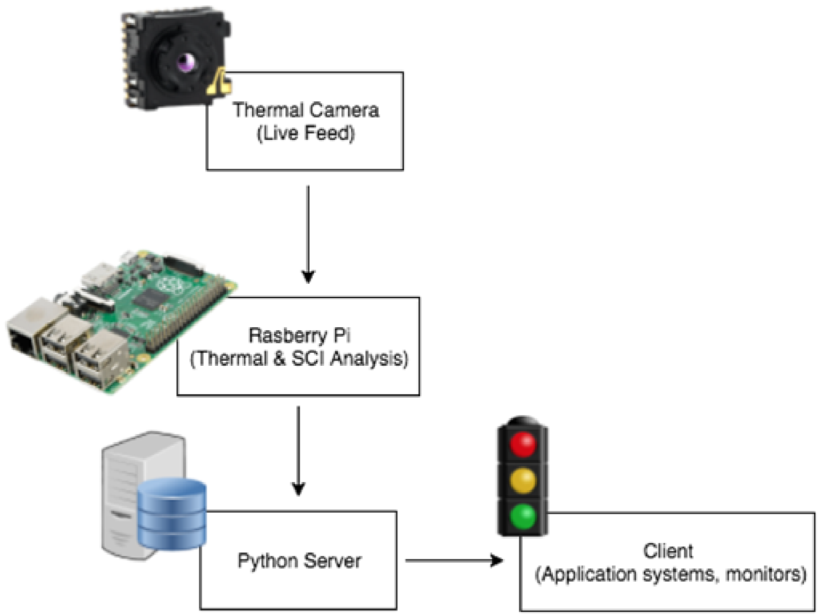

Fig. 10. Flowchart of feed to processor to server to client.

Thermal Camera Frame Analyzation The thermal camera takes pictures one after another to track movement of blob objects. Each frame is parsed through column by column, row by row using Python's Imaging Library.

```
from PIL import Image

allPixels = []

for y in xrange(im.size[1]):
        for x in xrange(im.size[0]):
                allPixels.append([x, y])
```

After each iteration of the function, the x value increases moving across the frame. A flood fill function is called recursively to each of the neighbors of the scanned pixels which are also in the threshold, once a pixel with a value out of range is found a new object is created and stored.

5.0.1 Thermal Camera Movement

Tracking Once each blob's midpoint is calculated, we can track the blob. When the next frame is captured, the previous midpoint's location is searched, if a midpoint is found within a certain proximity of the old midpoint's location, it is ID'd the same as the previous midpoint. Thus allowing Project Icarus to track midpoints, or "blobs".

5.1 Project Icarus Components

5.1.1 "FLIR Lepton" Thermal Camera

This micro thermal imaging camera works flawlessly in conjunction with the selected processing hardware.

5.1.2 Raspberry Pi

The Raspberry Pi is a compact, yet powerful computer that handles all of the image processing sent to it via the thermal camera.

5.1.3 Python Server

A simple Python WebPy server that displays the results of a the program. The results are fed to an HTML file which decodes the Base64 encoded image.

5.2 Applications

Several methods for controlling the movement of a group of people can be applied by the client once all the data is received from our program. Statistical analysis of crowds, funneling and management through lights or even people can be integrated seamlessly.

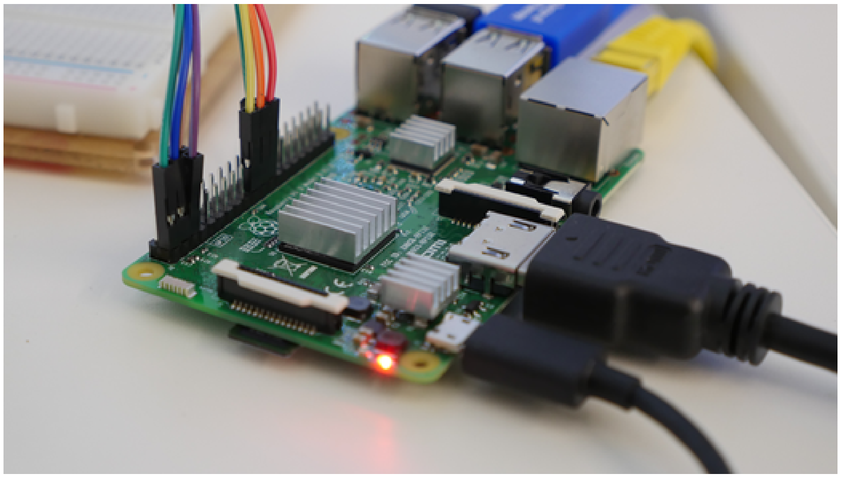

Fig. 11. Raspberry Pi.

5.3 User Scenarios

5.3.1 Crowd Control Aid

A peaceful protest ensues as citizens rally against a law recently passed by the city. As more people begin to gather, the city's crowd control team begins to monitor their walk across multiple streets. Icarus is launched, with camera placed at each intersection of interest. The system frequently collects instances of crowd movement, and after analyzation, displays the results on the server for all team members to view. After review of data retrieved, the crowd control team notices that the people are slowing down turning right on the street through their route. The team is then able to send members to that location in order to file them out in a more organized manner, in turn successfully migrating the crowd, and avoiding collateral damage.

5.3.2 Automated Crowd Control System

A concert is set to take place in a busy part of the city. Hundreds of people have lined up, and have just check their tickets in at the entrance. A crowd develops as everyone approaches the door to the musical hallway. Our software has already been set up to analyze the approaching crowd, and begins scanning people, grouping them into grouped sections based on the area they have taken up. The area is most likely entirely filled during the beginning, but as crowd begins to file through, Icarus is consistently searching for congested parts of the crowd. The system already holds a map of the entrance, and has a calculated route that would serve as the most efficient way to walk the crowd through the doors. It signals to individual sides of the crowd to adjust their speed (or direction if need be), therefore allowing other sections of the crowd to pass through easily and quickly.

5.4 Facilitated VS Un-Facilitated Funneling

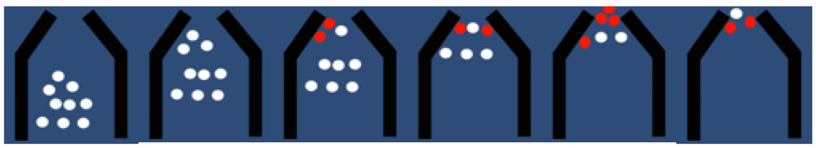

Fig. 12. Facilitated.

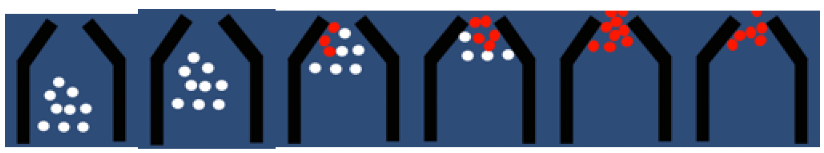

Fig. 13. Un-Facilitated.

6 Conclusion

By analyzing the properties of a crowd and identifying congestion points, our software can now reform crowd management. By efficiently scanning the proximity, planning an optimal route, and delivering signals for people to follow, Stampedes, congestion, and traffic can now be prevented. With Project Icarus capable of functioning as an individual unit, or as an aid for a crowd control team, this device will always be a supplement to making crowds safer and faster to travel in.

Continued work and updates can be seen on http://www.projecticarus.io/ and https://www.github.com/SaranshPK/HumanTracking.

References

1. http://www.huffingtonpost.com/entry/hajj-stampede-death-toll_us_5625d226e4b0bce347020aa1
2. http://www.cnn.com/2015/09/25/middleeast/hajj-pilgrimage-stampede/