Inspection Robot with Low Cost Perception Sensing

Harriet Peel*, George Morgan, Colin Peel, Anthony Cohn, **Raul Fuentes,
School of Civil Engineering, University of Leeds, Leeds, LS2 9JT
*Email addresses *cnhap@leeds.ac.uk, **r.fuentes@leeds.ac.uk*

Abstract - This paper presents the use of off-the-shelf products as a low cost solution to bridge bearing inspection. A commercial product, known as a DiddyBorg, is a robot designed for use with a Raspberry Pi as the on-board computer. The DiddyBorg is used as a robotic platform to make a photogrammetric survey of the bearing area of a bridge. The images collected from this survey are then used to make a 3D reconstruction using Structure-from-Motion (SfM) and software 3DFlow Zephyr Aerial (Zephyr). The quality of the 3D reconstruction had an accuracy of +/- 30 mm when compared to the known dimensions of the area. The resulting point cloud was then used as a map that the robot can use for navigation purposes. In particular we present a simple localization algorithm based on distance three readings measured from the robot.

Keywords-
Structure-from-Motion, robot, inspection, point cloud

1 Introduction

Bearings are critical for the performance of bridges and yet surveys are carried out infrequently, in many cases due to difficult access.

A solution to this problem is the use of autonomous robots which allows carrying out inspections with more repeatability than the traditional visual inspections that engineers undertake.

In particular, the geometry of the bridge bearing is of great importance. Although this paper does not focus on a methodology to achieve high accuracy, we present the use of a commercially available robot that uses low-cost cameras to undertake a full 3D reconstruction of the bearing area using Structure-from-Motion (SfM). There have been several applications of SfM to geosciences [1], archaeology [2] and for assessing the progression of work on construction sites [3]. One advantage of using SfM is that data collection using digital photography already occurs in many instances. Therefore, it is possible to use the images that have already been collected for other purposes. The images collected by the camera for the reconstruction can also be used to make a record of the status of the bridge and to observe damage using other techniques such as digital image correlation.

The focus of this paper is on the important issue of autonomous navigation. Specifically we present a solution to localization in near real-time using distance measurements and the initially recorded point cloud.

2 Structure-from-Motion (SfM)

SfM is now a relatively mature 3D reconstruction technique, much literature exists on the subject, and several commercial products exist. Zephyr Aerial is one such commercial software made by 3DFlow [4] and uses a proprietary algorithm known as SAMANTHA [5].

SfM uses multiple 2D views to find the 3D geometry (i.e. the structure) of a scene or an object by taking a series of images from different viewpoints (i.e. the camera has motion). The images collected do not need to be organised/ordered, nor do the camera locations need to be planned. Distinctive features, known as key points, are extracted from the images using feature detectors such as the scale-invariant feature transform (SIFT) [6], or in the case of SAMANTHA, a feature detector with automatic scale selection based on the work by Lindeberg in [7].

In SAMANTHA, keypoints are then matched between images in two stages: broad stage matching and narrow stage matching [5]. Broad stage matching uses keypoints that are ranked according to the scale value obtained in keypoint detection: keypoints with the strongest response are used for matching with keypoints in other images. Once they are matched between images, the corresponding images are connected to the images with which it shares the greatest number of keypoint matches to form the epipolar graph. The epipolar graph describes how views from different cameras are geometrically related.

In narrow stage matching, some keypoints are discarded depending on nearest neighbour relationships. The fundamental matrices, which describe the location and rotation of the cameras in a global (but unscaled) reference frame, are then calculated using a subset of the key point matches. To find the fundamental matrix between two camera views, at least 7 keypoints are required. The fundamental matrix is commonly

calculated using RANSAC [8], although SAMANTHA uses a variation of RANSAC called M-estimator sample consensus (MSAC) [9] which applies a set penalty to outliers.

Once the fundamental matrix has been calculated, the relative positions of all the cameras can be computed. Some SfM methods [10] use an incremental approach, adding one new camera in at a time and using triangulation to find the 3D geometry, then bundle adjustment is carried out using Bundler [11]. SAMANTHA uses a 'hierarchical' approach [5]. The advantage of the hierarchical approach is that the calculations are parallelisable. Initially, images are clustered based on the amount of overlap between them. Partial stereo models are formed between the individual clusters of images. These partial stereo models can then be merged with other partial models, or can be expanded with other individual images. Bundle adjustments are carried out at each stage. Local bundle adjustment may also be used [5].

It is important to note that for all SfM methods the absolute scale of the scene cannot be recovered and that some method of scaling is required to relate the point cloud to a global scale.

A technique known as Multi-View Stereo [12], [13], can then be used to enhance the point cloud generated using SfM to make a dense point cloud.

3 Robot Description

A robot was selected to fill the following criteria;

- To be small and compact enough to navigate through limited spaces.
- Could be programmed.
- Could be adapted for autonomous navigation.
- Could be controlled remotely.
- Could capture photographs of a suitable standard for use in 3D reconstructions.

A variety of models were considered that fulfilled these criteria, and the model selected was the DiddyBorg, a self-build kit made by PiBorg [14]. The DiddyBorg is a six wheeled vehicle approximately 182 x 220 x 95mm. Each wheel is driven by 5V motor, which is required to be used outside and required an adequate level of torque. The DiddyBorg is built around a Raspberry Pi Model 2 B motherboard with the operating system Raspbian Jessie [15].

The DiddyBorg with a Raspberry Pi was chosen for this application due to its low cost (approx. £180 / $260) and ease of programming as it is open source. The Raspberry Pi adds the ability to mount multiple sensors. For the experiments carried out in this paper, three ultrasound sensors were mounted to the DiddyBorg, one at the front and one on the left and right sides of the DiddyBorg. A further description of these sensors is provided in Section 7.

The camera used is a Raspberry Pi Camera Board [16], it produces 5MP pictures and 1080p HD video at 30fps. The software required to control the camera is also open-source and readily available.

4 Site Description

The site considered in this paper is the Centenary Bridge, in south Leeds, UK, opened in 1993. It crosses The River Aire, connecting The Calls to Brewery Place. It is a cable suspension bridge spanning approximately 57m. The bridge was chosen because it was easily accessible and the bearings are located in an enclosed region (Figure 1) in which the DiddyBorg system could be tested easily. One disadvantage of this area being accessible, is that there was a build-up of litter inside the bearing enclosure, the quantity of this litter is likely to change from inspection to inspection and cause anomalies in the point clouds that are generated. However, not being able to control the environment highlights one advantage of using camera as a sensor over methods such as laser scanning – a visual assessment can easily be made of the quality of the data that is obtained.

The bearings in question are the north side bearings. The maximum height of the bearings and subsequently of the enclosure is 400mm. The top bearings are seated on the bridge by means of a machined steel plate bolted to the bearing. The bottom bearings are seated on the abutment by means of bedding mortar.

The top of the abutment is a space approximately 2.8m by 1.2m (Figure 3). There are few restrictions to the site. Firstly, there is a 155mm wide trough that runs along one side on the site. This is too wide for the robot to cross so the robot's movements were restricted to one side of this trough. Secondly, there were various pipes and electrical cables that occupied the centre of the site. The robot could travel over these obstacles, but with risk of getting stuck, so these regions were avoided. There was also a gate at street level that had a top support within the site, which was avoided.

5 Survey

For reconstructions of enclosed areas, such as a room or a town square, the best practice, as recommended by [17], is to take a panoramic sequence of images from at least 2 corners of the region being captured and to obtain at least 60% overlap between images. To meet these recommendations, the DiddyBorg platform was rotated by small increments up to 360 degrees in three locations – refer to Figures 4 and 5. Each time the DiddyBorg was

rotated and stopped a photo was captured. Data capture took in the region of 20 minutes to complete.

5.1 Reconstruction

The photographs were then reconstructed using Zephyr Aerial. Due to computational requirements, and since reconstruction on-board the robot was not necessary, the reconstruction was performed on a desktop PC. Since Zephyr Aerial produces a reconstruction automatically from the data it is given, no changes in the default settings were made to the Structure from Motion or Multiview stereo stages of the reconstruction. The reconstruction took approximately 20 minutes running on a desktop computer with 32.0 GB RAM, a 3.60 GHz CPU and two 2.0GB NVIDA video cards with CUDA capabilities. Cloud computing services for reconstruction are also commercially available from 3DFlow [18].

There was some noise from the changing lighting conditions and the litter, and for this reason a confidence calculation was performed in Zephyr. The confidence score is calculated by considering how well a point is matched between photos. The matching score is from 0 to 1 and is summed over the photos. The confidence calculation in Zephyr removes points where a certain matching score has not been reached. In this case, points with a confidence value less than 1 were removed. The resulting point cloud, after updating the confidence, is presented in **Error! Reference source not found.** and Figure 4.

Since SfM creates a point cloud at an arbitrary scale, a method is required to scale the point cloud. Control points were picked from the photographs. The point cloud was then scaled using the scale drawings from the bridge design, this is a feasible approach for applications in inspection of civil infrastructure.

In this case, four control points were used, more control points would give a more accurate reconstruction, but since the environment surveyed here is broadly of one texture it was difficult to find points that clearly stood out that also corresponded to the scale drawings. For this reason, points such as corners were used. Once the control points are picked (control points appear as dots marked in **Error! Reference source not found.** and Figure 4), a further bundle adjustment is carried out and an RMS error is then calculated by Zephyr showing the difference between the values used and their locations within the Zephyr model. The RMS error for this scaled reconstruction was 30mm.

Figure 1: Photograph showing the location of the north-side bearings of Centenary Bridge, Leeds, UK.

Figure 2: Photograph showing the enclosure in which the bridge bearing survey was carried out using the DiddyBorg robotic platform.

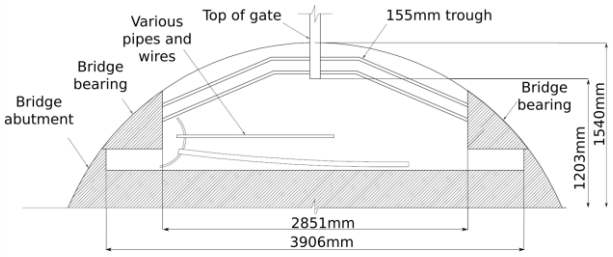


Figure 3: Schematic showing the dimensions and features of the enclosure from Figure 4.

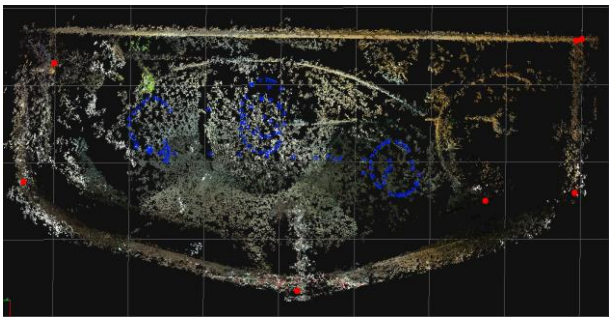


Figure 4 : Top view of the SfM reconstruction of the bridge bearing enclosure generated using Zephyr Aerial with confidence of points updated. Control points are marked by red dots. The blue shapes in the middle of the cloud show the locations where the robot took pictures.

5.2 Platform Integration

To interface with the Raspberry Pi, a network development environment known as Node.js [19] was used. The open-source nature of the community developing with Node has led to the development of many exciting projects, and many libraries exist that can easily be installed and integrated into a project. The work here to integrate the Raspberry Pi with the DiddyBorg robotic platform with several different sensors uses and brings together work from libraries including: r-pi-usonic [20], picoborgrev [21] as well as taking some inspiration from the web Graphical User Interface developed by PiBorg [22] specifically for manipulating the motion of the DiddyBorg.

Requests were sent from a webpage at the front end to a web application program interface (API) which interfaces with the Raspberry Pi. Commands are sent to control the motors and the camera, but this interface is easily expandable to incorporate any other sensors that are on-board. The algorithm for detecting the location of the DiddyBorg using distance measurements was also integrated into this system and the results can be viewed directly from the webpage. The design criteria when creating this system was to use familiar technology, like a webpage, for ease of use when working in the field. The user can manipulate the motion of the robotic platform, take video footage or camera stills, distance measurements and calculations from on-board sensors (e.g. ultrasound, infrared) and calculate the location of the platform through a simple and straightforward interface.

6 Algorithm

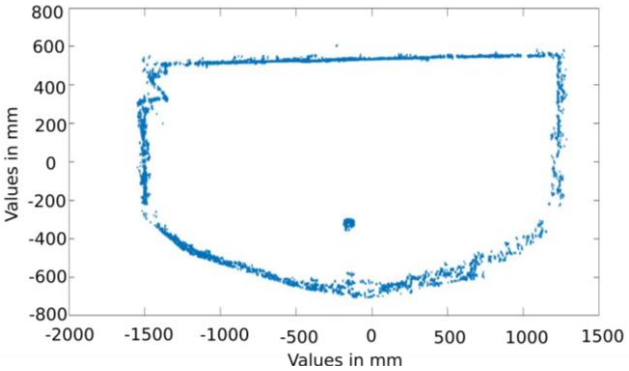
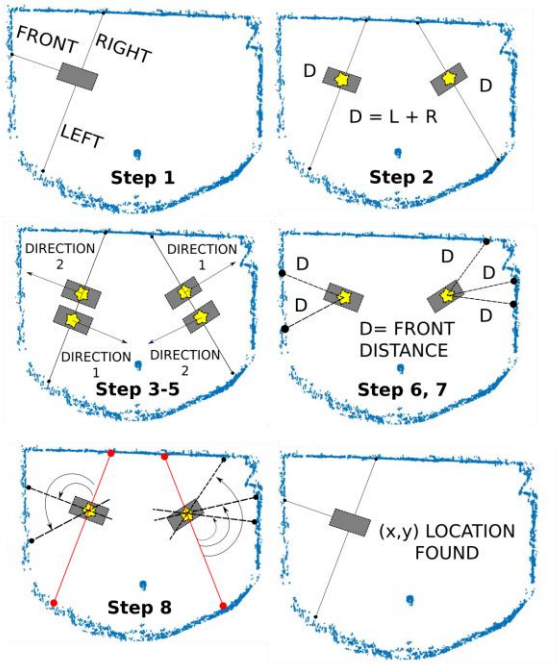


Figure 5: Convex hull point cloud generated using Cloud compare (plotted here as (x,y) coordinates with values in mm).

Once a point cloud had been generated, it had to be reduced in a way that would be useful for the navigation of the robot. The proposed solution was to produce a convex hull of the point cloud (see Figure 5), as an outline of the region the DiddyBorg is in. This outline was taken at the same height as the sensor mounts on the DiddyBorg. This reduction was completed using CloudCompare [23]. A localisation algorithm was then required to obtain the location of the DiddyBorg within the region using distance measurements. The steps of the localisation algorithm are as follows (also see Figure 6):

1. Take a reading from each of the sensors: front, left and right.
2. Find pairs of points where the distance between the points is equal to the sum of the left and right readings within a given error.
3. Using trigonometric relations and similarity find a guess location for the location of the DiddyBorg.
4. This method cannot calculate which direction the DiddyBorg is facing, so
5. Guess locations for two directions must be considered.
6. Calculate the distance from each guess point to all other points in the point cloud.
7. Find whether any of the distances match the front sensor reading within a given error.
8. Narrow down the number of possible locations by calculating the gradient of the lines connecting the points joining the side measurements and the gradients connecting the guessed point and the points that match the front readings. If the gradient of one is equal to -1/gradient of the other then the points are perpendicular, this is a requirement since the sensors are perpendicular on the DiddyBorg platform.



=

Figure 6: Schematic showing the steps of the algorithm described in Section 6.

It is highlighted that this algorithm will not give the unique location of the DiddyBorg platform when the point cloud used for navigating has axes of symmetry. However, the aim at this stage of the project was to reduce the possible number of locations the robot could be. Improvements to this algorithm and the methods used to detect potential locations will be the target of future work.

7 Validation

The algorithm was validated from tests within a simulated rectangular convex hull point cloud. Test locations for the DiddyBorg were chosen inside this location, the corresponding sensor readings were calculated and the algorithm was run.

A rectangular area was chosen to test the algorithm because it is a case where multiple locations will be returned due to symmetry. If the algorithm was successful, it was expected that four predicted locations would be returned.

Within the algorithm, (listed above in Section 6) an error value is used to account for the error in the sensor readings. This value is a constant and can be varied. When the error value is set to be very low (below 0.05 mm), i.e. the accuracy of the sensor is very high, the algorithm returns 4 possible locations for the DiddyBorg including the test value set at the beginning and showed that the logic of the algorithm is correct. However, this error value is unrealistic for what can be expected from a real sensor, and as the error value was increased more possible locations for the robot were detected, such that it would not be possible to say for certain where the robot was.

To determine the reliability that might be expected from using real sensors, an ultrasound sensor HC-SR04 Ultrasonic Module Distance Measuring Transducer Sensor [24]) that returns distance from an object was tested. The sensor range from the sensor datasheet [24] is 2cm to 4m, with an accuracy of 3mm. This sensor has been used with the Raspberry Pi [25] and the Arduino [26] open-source electronics platform.

The ultrasound sensors were wired in a similar way to [25], but by daisy chaining the ultrasound sensors from the battery pack that also powers the Raspberry Pi and the motors. The system and user interface developed in Section 5.2 were easily adapted to incorporate the functionality of these sensors. The algorithm used to take readings from the sensor was based on the work of [20], expanding it to incorporate multiple sensors. The sensors were tested by taking readings when the robot was stationary and then checking the readings against manual measurements using a tape measure. When the robot was facing a large planar surface, such as a wall at medium range, the readings were consistent and within the accuracy stated in [24]. However, the distance readings from the sensor became unreliable with variable surfaces and obtructions with readings fluctuating by up to 20cm in some cases. This variability is likely due to measurement angle of the sensor being 15 degrees, so a different object to the one the ultrasound sensor is facing may have been detected. Therefore, if this sensor was to be used with the algorithm in Section 6 it would, at best, be able to slightly narrow down the number of potential locations, but not adequately enough for navigation purposes. It was concluded that this sensor is inadequate for use on its own for the purpose of navigation, but perhaps if localisation is carried out in with other sensors, the ultrasound sensors could be used to detect unexpected objects such as the litter found in the bridge bearing enclosure.

8 Future Work

Given that the ultrasound sensors were not accurate enough to use for navigation purposes, in the real case of the bridge bearing enclosure, alternative sensors will be identified and tested for the purpose of navigation. Further work will also be carried out to investigate whether the ultrasound sensors can be used effectively to detect unwanted objects in the region being inspected such as litter.

In order to deal with uncertainties encountered in the

field, further development of the algorithm presented in Section 6 will be carried out.

A method for independently scaling point clouds without the use of scale drawings (although the drawings may be used as a measure of accuracy) is to be developed. This method should be easily integrated into the surveying process detailed in Section 5.

9 Conclusions

The DiddyBorg platform was used to successfully survey a bridge bearing through photographic data collection. These photographs were then used to reconstruct the enclosure where the bearings are situated using the SfM software Zephyr Aerial. The point cloud genereated in this reconstruction was used to create a convex hull point cloud suitable for use with the localisation algorithm developed for navigation (Section 6), which, in principle, can be used with the DiddyBorg robotic platform that was used to gather the original photographic data. The robot was successfully controlled using the web-based API detailed in Section 5.2.

10 Acknowledgements

We would like to acknowledge funding from ESPRC for the studentship of H.Peel.

References

[1] J. Westoby, M. Brasington, M. Glasser, N. Hambrey and M. Reynolds, "Structure-from-Motion' photogrammetry: A low-cost, effective tool for geoscience applications," *Geomorphology,* vol. 170, pp. 300-314, 2012.

[2] C. Balletti, F. Guerra, V. Scocca and C. Gottardi, "3D Integrated Methodologies For The Documentation and the Virtual Reconstruction of an Archaeological Site," in *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XL-5/W4*, Avila, Spain, 2015.

[3] M. Golparvar-Fard, F. Peña-Mora, S. Savarese and e. al, "Automated Progress Monitoring Using Unordered Daily Construction Photographs and IFC-Based Building Information Models," *Journal of Computing in Civil Engineering,* vol. 29, no. 1, p. 147, 2012.

[4] 3DFlow, "3DFLOW," 3Dflow, 2016. [Online]. Available: http://www.3dflow.net/. [Accessed 03 04 16].

[5] R. Toldo, R. Gherardi, M. Farenzena and A. Fusiello, "Hierarchical structure-and-motion recovery from uncalibrated images," *Computer Vision and Image Understanding,* vol. 140, pp. 127-143, 2015.

[6] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision,* vol. 60, pp. 91-110, 2004.

[7] T. Lindeberg, "Feature detection with automatic scale selection," *International Journal of Computer Vision,* vol. 30, pp. 79-116, 1998.

[8] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Communications of the ACM,* vol. 24, pp. 381-395, 1981.

[9] P. H. S. Torr and A. Zisserman, "MLESAC: A new robust estima-tor with application to estimating image geometry," *Computer Vision and Image Understanding,* vol. 78, no. 1, p. 138–156, 2000.

[10] N. Snavely, S. M. Seitz and R. Szeliski, "Photo tourism: exploring photo collections in 3D, in: SIGGRAPH," in *International Confer-ence on Computer Graphics and Interactive Techniques, pp 835–846*, 2006.

[11] M. Lourakis and A. Argyros, "SBA: a software package for generic sparse bundle adjustment.," *ACM Transactions on Mathematical Software,* vol. 36, p. 1–30, 2009..

[12] Y. Furukawa and J. Ponce, "Accurate,dense, and robust multi-viewstereopsis," in *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, USA, 2007.

[13] Y. Furukawa, B. Curless, M. Seitz and R. Szeliski, "Clustering view for multi-view stereo," in *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, USA, 2010.

[14] PiBorg, "DiddyBorg," 03 April 2016. [Online]. Available: https://www.piborg.org/diddyborg.

[15] R. P. FOUNDATION, "Download Raspian for the Raspberry PI," [Online]. Available: https://www.raspberrypi.org/downloads/raspbian/. [Accessed 03 04 2016].

[16] R. P. FOUNDATION, "Raspberry Pi Camera Module," [Online]. Available: https://www.raspberrypi.org/products/camera-module/. [Accessed 03 April 2016].

[17] 3Dflow, "3DFLOW," 2016. [Online]. Available: http://www.3dflow.net/technology/documents/photogrammetry-how-to-acquire-pictures/. [Accessed 03 04 2016].

[18] 3DFlow, "SUNNY Beta - the Cloud service for your reconstruction," [Online]. Available: https://sunny.3dflow.net/login.php. [Accessed 03 April 2016].

[19] Node.js.Foundation, "node JS," 2016. [Online]. Available: https://nodejs.org/en/. [Accessed 03 April 2016].

[20] clebert, "A high performance, memory mapped, Node.js API for the HC-SR04 ultrasonic sensor connected to a Raspberry Pi.," 2015. [Online]. Available: https://github.com/clebert/r-pi-usonic. [Accessed 01 April 2016].

[21] CodyErekson, "picoborgrev: A controller library for PiBorg's PicoBorg Reverse motor driver board designed for use with the Raspberry Pi.," 2014. [Online]. Available: https://github.com/CodyErekson/picoborgrev. [Accessed 03 April 2016].

[22] PiBorg, "diddyborg-web: Web based interface for controlling the DiddyBorg from a phone or browser," 2015. [Online]. Available: https://github.com/piborg/diddyborg-web. [Accessed 15 March 2015].

[23] D. Girardeau-Montaut, *Cloud Compare—3D Point Cloud and Mesh Processing Software*, Open Source Project - http://www.danielgm.net/cc/, 2015.

[24] E. Freaks, "Ultrasonic Ranging Module HC - SR04 Product Features," [Online]. Available: http://www.micropik.com/PDF/HCSR04.pdf. [Accessed 03 04 16].

[25] P. Staff, "Building a Raspberry Pi Motion Sensor with Realtime Alerts," PubNub, 16 June 2015. [Online]. Available: https://www.pubnub.com/blog/2015-06-16-building-a-raspberry-pi-motion-sensor-with-realtime-alerts/. [Accessed 03 April 2016].

[26] Arduino, "Genuino," 2016. [Online]. Available: https://www.arduino.cc/. [Accessed 03 April 2016].

[27] R. P. FOUNDATION, "Raspberry Pi - Teach Learn and Make with Raspberry Pi," [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/. [Accessed 01 04 2016].

[28] Itseez, "OpenCV," 2016. [Online]. Available: http://opencv.org/. [Accessed 04 April 2016].