THE INSTITUTE FOR SYSTEMS RESEARCH

ISR TECHNICAL REPORT 2014-07

Examining Vision Sensor for Cm-Scale Robots

M. Frantz, I. Penskiy, and S. Bergbreiter

Examining Vision Sensor for Cm-Scale Robots

M. Frantz, I. Penskiy, and S. Bergbreiter, Member, IEEE

***Abstract*—The capabilities of the Centeye Stonyman Vision Chip were examined on the Parallax BOE-bot platform. This paper describes the tests that were performed to determine the sensor's suitability for controlling navigation in an unknown environment. M. Srinivasan's Image Interpolation Algorithm was used to compute 1D optical flow. Parameters that affect the optical flow's signal-to-noise ratio were manipulated to acquire a sufficiently strong signal and navigation of a testing arena and collision avoidance were demonstrated.**

Keywords—robot vision; miniature robot; obstacle avoidance

I. INTRODUCTION

At the International Society for Optics and Photonics Conference in 2000, Douglas Gage provided a manifesto on the future of distributed robotics research [1]. He pointed out that for the robotics community to actualize groups of networked robots that perform meaningful tasks, such as navigation and mapping, the robots must necessarily meet three requirements: they must be inexpensive, easy to manufacture and durable. Since his writing, many in the distributed robotics community have developed centimeter-scale autonomous mobile robot platforms to meet these three requirements. EPFL has developed Alice [2] and has demonstrated Alice's use in adaptive behavior experiments with multiple robots [3]. Harvard recently introduced Kilobot in order to study the algorithms and control methods that apply to a large collective of decentralized robots [4]. On the home front, TinyTeRPs have been developed at University of Mayland to explore collaborative behavior [5].

However, one challenge to the variety of tasks that these centimeter-scale robots can perform results from the lack of available resources with respect to robust environmental sensing. Electrical and computational power constraints deny small-scale robots access to various robust localization and mapping setups. Research in the area of low-power sensors that require minimal computation to provide sufficient data for navigation is therefore essential. EPFL's Alice uses IR sensors to detect obstacles, but these obstacles must be in close proximity to it [2]. Kilobot has very limited IR sensing that allows it to detect the presence of other Kilobots within a 10 cm radius [4]. Similarly, TinyTeRPs have made use of radio signal strength to detect proximity to each other [5]. None of these robots has any sort of long range environmental sensing that allow for navigation of an unknown environment.

Long-range environmental sensing can be accomplished with various devices, such as laser rangefinders, sonar sensors and cameras. Of these three, cameras provide the most flexibility with respect to the tasks they can perform. One example use is for communication and localization of multiple robots through light tracking [6]. Cameras can also be used as optical flow (OF) sensors in order to avoid obstacles with minimal computation. For example, Zufferey and Floreano describe the use of two 1D cameras to control navigation of a 30-gram airplane with on-board computation [7]. Zufferey and Floreano have also demonstrated the use of a single wide-angle camera for enabling collision avoidance on the wheeled Khepera robot [8], which provided a majority of the inspiration for this work. The goal of this research is to demonstrate the use of a single camera for controlling the navigation of small wheeled robots through an unknown environment of arbitrary geometry.

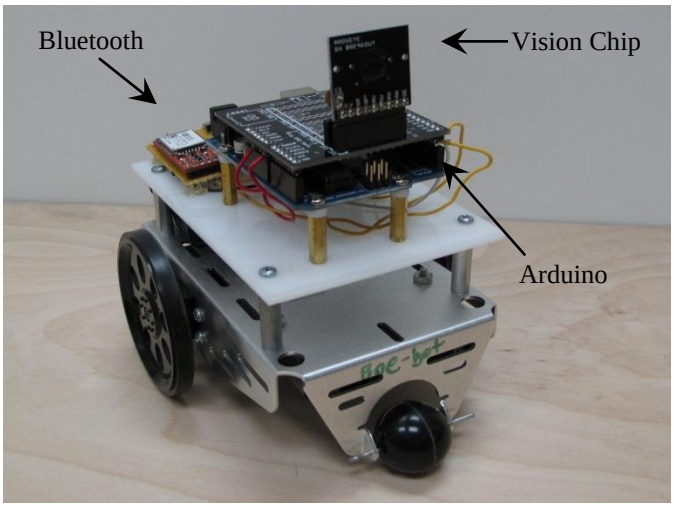

Fig. 1. Parallax BOE-Bot with Arduino Uno, Centeye Stonyman Vision Chip and Sparkfun Bluetooth Mate.

To do this, the Centeye Stonyman Vision Chip was chosen for its size, its relatively inexpensive "off-the-shelf" status and for support from Centeye that facilitated the ability to experiment with the sensor. This support included the chip's packaging in a breakout board, its easy connection to an Arduino via the Stonyman Uno Rocket Shield, and the open-source software support available on ArduEye.com. In order to study the capabilities of the chip, it was mounted on the Parallax BOE-Bot platform while data was collected from an Arduino Uno (Fig. 1). This paper describes the testing that was done to characterize the sensor's capabilities and discusses the results of these tests. While there are many uses of an onboard vision sensor, e.g. localization and mapping [9] and light tracking, this paper focuses on the use of OF for collision avoidance. The rest of this paper is organized as follows: Section II provides an overview of OF and its previous use for collision avoidance. Section III describes the testing platform and environment. Section IV explains the experiments that were performed with the sensor and the results of these experiments. The paper culminates with Sections V and VI providing ideas for future work and acknowledgments.

Fig. 2. Randomly generated textured wallpaper for OF testing.

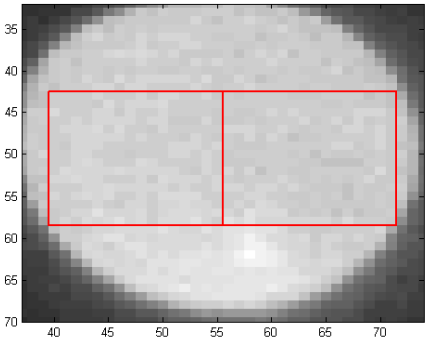

Fig. 3. Active pixels on the pinhole camera. Red boxes show position of 16x16 pixel subwindows.

II. OPTICAL FLOW

As a camera moves around a 3D environment, it captures 2D projections (images) of this environment. In a given duration of time, points in the 3D space will move. This results in a related motion of points in the images. The 3D velocity vectors that characterize the motion of points in space are referred to as the motion field. The 2D motion field is made up of the projections of these 3D vectors onto the image plane [10]. It is this 2D motion field that is useful in localizing the camera with respect to points in its environment.

OF is the apparent motion of an image from one frame to the next. It can be estimated by comparing pixel intensities between consecutive frames. This comparison is used to estimate the motion field of the environment. Because OF can be relatively computationally efficient and robust to noise, OF has been used by many groups as a means of navigational control of a robot through an environment [7-8,11-13]. Although outdoor environments tend to have sufficient texture for the use of OF [13], these examples take place in indoor testing arenas. Due to the limited texture of indoor environments, in most of these examples the boundaries of the testing arenas consist of some contrasting black-and-white imagery on the walls. Sufficient texture is required due to the fact that OF compares light intensities between pixels—in an environment whose boundaries tend to have smooth, monochromatic texture, the camera will tend not to be able to distinguish one pixel from another. For this reason, all tests reported in this paper make use of a wallpaper of randomly arranged black and white squares, as shown in Fig. 2. The real-life size of each square shown in the figure is one cm^2.

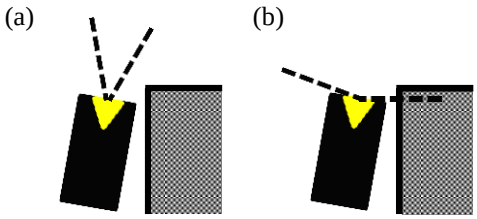

Fig. 4. Depiction of robots with different FOV's as they drift toward an obstacle. (a) shows the lens (~40° FOV). (b) shows the pinhole (~160° FOV).

Many algorithms have been developed to compute OF. The experiments discussed in this paper use the Image Interpolation Algorithm developed by M. Srinivasan [14] and implemented for computing 1D OF on Arduino by Centeye, Inc. This algorithm was selected due to its computational efficiency and due to its demonstrated performance on other navigating robots [7-8,10,14].

III. TESTING APPARATUS AND ENVIRONMENT

The sensor that was studied was the Centeye Stonyman Vision Chip, consisting of a 112x112 array of greyscale pixels. Two kinds of optics for the chip were explored: a wide field-of-view (FOV) pinhole and a lens with a 4mm focal length. The wide FOV optics covers all but an approximately 35-pixel diameter so that only these uncovered pixels are active. This allows for an approximately 160° FOV. The exact location of the active pixels was determined by taking an image of a brightly lit white background and subtracting fixed-pattern noise (Fig. 3). The lens optics has a much narrower FOV (approximately 40°), but all pixels are active. Due to memory limitations of the Arduino Uno, only a portion of the active pixels was used to calculate OF. This portion was always a rectangular array (subwindow) determined with reference to the center of the active pixels. After initial testing, it was determined that although the lens optics provided a cleaner OF signal, the narrow FOV restricted its ability to successfully avoid collisions with the environment. Fig. 4. illustrates this. 4a shows that the robot with a narrower FOV may move toward an obstacle at a shallow angle of impact without being able to detect it, while 4b shows that the larger FOV should more readily detect the obstacle. In keeping with the goal of using a single camera to navigate and avoid collisions, the pinhole optics was chosen for further exploration. For this reason, only the results from the pinhole optics are discussed in section IV.

The sensor was mounted using a Centeye Rocket Shield plugged into the Arduino Uno. This setup was mounted on the Parallax BOE-bot platform (Fig. 1). The distance between the sensor and the servo axle was 6 cm. Data was streamed serially using either USB or a Bluetooth connection.

Initial testing was performed in a square arena measuring 1 meter per side. The walls of the arena were covered with a 50 centimeter tall wallpaper of randomly arranged black and white squares described above. Final testing was performed in an arena of arbitrary geometry with the same wallpaper.

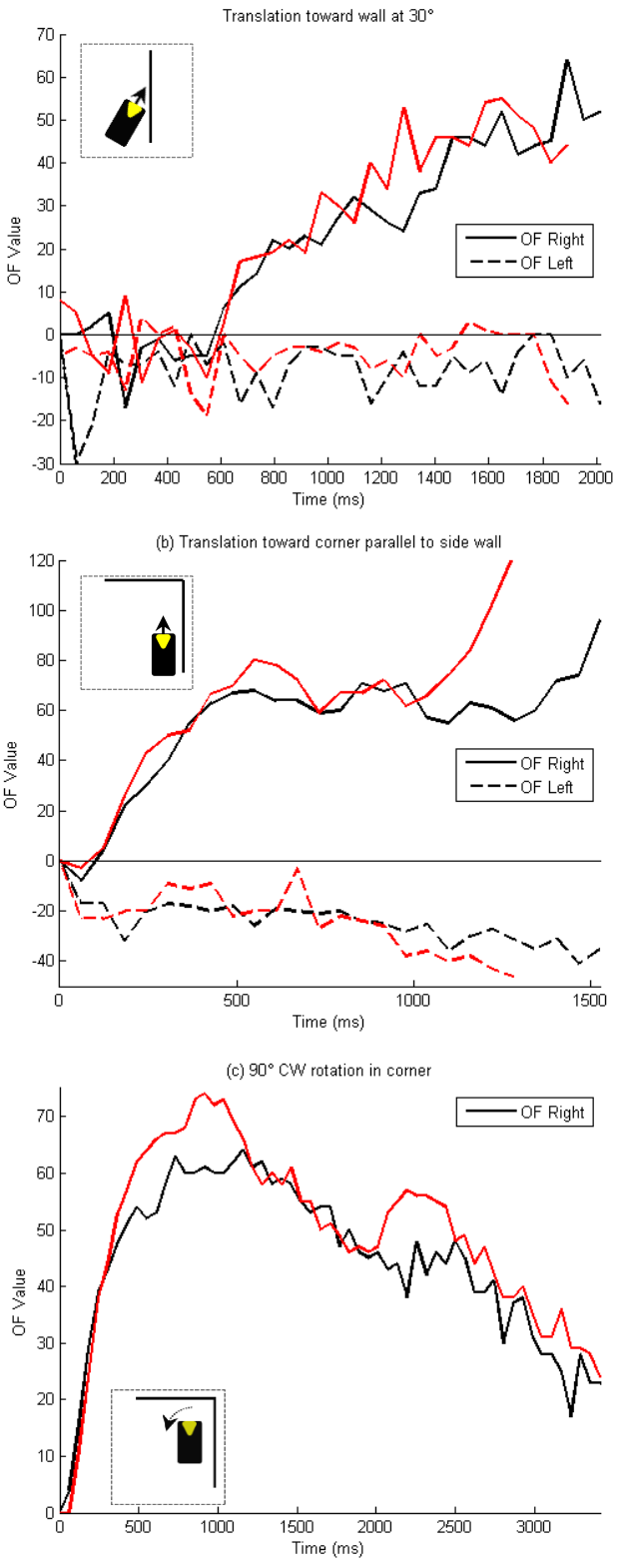

Fig. 5. OF signals during three scenarios the robot encounters during navigation of the arena. (a) shows data collected during a translation toward a wall at an angle of 30°. (b) shows data collected during translation toward a corner as shown. (c) shows data collected during a CCW rotation of 90° in a corner, starting in the position shown in the inset. Translation speeds were ~18 cm/s. Angular velocity during rotation was ~45 °/s. Two trials are shown for each scenario (black and red).

IV. EXPERIMENTS

To achieve the goal of using OF to navigate around the arena while avoiding collision with walls, a modified version of the control algorithm described by Floreano and Zufferey was used [8]. This algorithm calls for two types of motion: translation and rotation. During translation, the control algorithm makes use of two OF values, one from the right side of the camera (OF Right), the other from the left (OF Left). These values are computed from the subwindows shown in Fig. 3. As will be detailed later, the algorithm has been modified to make use of a single OF value during rotation.

A. Obstacle Detection Scenarios

To prepare for use of the control algorithm, several tests were performed. It was discovered early on that the signal-to-noise ratio (SNR) was dependent on several parameters, including the size and position of the subwindows, the ratio of sampling frequency to speed, and the feature size and configuration of the textured wallpaper described above. While these parameters were never optimized, they were sufficiently manipulated to achieve an adequate SNR for reliable control of the robot.

In practice, horizontal 1D OF was computed by capturing a subwindow, subtracting fixed-pattern noise, summing the pixel intensities of each column of the subwindow, then comparing the resulting 1D array of intensities to the previously computed 1D array using Srinivasan's Image Interpolation Algorithm [14]. The size of the subwindow was limited both by the number of active pixels on the pinhole camera and by the memory capacity of the Arduino Uno. Testing showed that the OF signal with the greatest SNR was found using 16x16 pixel subwindows located as shown in Fig. 3.

When both OF Right and OF Left were computed during translation, the frequency for computing OF from the two 16x16 subwindows was 16 Hz. During translation, slower speeds resulted in a noisier signal. The best signal obtained during testing occurred when the robot's average speed was 18 cm/s. When only one OF value was computed during rotation, the sampling frequency was 33 Hz. It was found that during rotation, the OF SNR would decrease as the angular velocity increased. The best signal obtained during rotational testing occurred when the robot rotated at about 45 °/s.

Fig. 5 shows plots of OF data collected for three different scenarios that are pertinent to the robot's navigation of the arena. 5a shows the OF Right and OF Left as the robot approaches a side wall at an angle of 30° from parallel. The OF Right signal shows an increase in magnitude, while the average OF Left signal remains unchanged. 5b shows the OF Right and OF Left as the robot approaches a corner as illustrated in the inset. The greater change in magnitude of the OF Right signal, as in 5a, indicates the ability to distinguish in which direction the robot should turn.

Fig. 5c shows the OF Right signal as the robot rotates 90° CCW in a corner from the illustrated starting position. This signal changes as a result of the sensor's distance from the axis of rotation and its changing distance from the walls during rotation. The OF magnitude reaches a peak when the

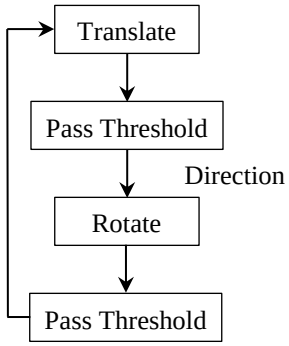

Fig. 6. Control algorithm for proportional-turn collision avoidance.

camera is at the shortest distance from the wall, i.e. when the velocity at which the wall passes in front of the camera is maximal. The change in OF indicates the ability to determine a point at which the robot is close to parallel to the side wall.

Similar data was collected for the symmetrical cases of approaching a wall on the left, approaching a corner on the left and rotating CW in a corner. In each case, the OF signals provided sufficient information for the robot to be able to identify when and how it should turn.

B. Proportional-turn Collision Avoidance

The control algorithm used to navigate through different environments is shown in Fig. 6. The robot begins by translating forward while collecting OF Right and OF Left data. Due to signal noise, the rotation condition was set such that the OF value must exceed the threshold value a number of times consecutively, at which point the robot will rotate in place. This number was determined empirically. The direction of rotation is determined by which signal's threshold is passed. As opposed to translation, where the signal passes the threshold from low to high, during rotation the signal passes the threshold from high to low. Once the rotation threshold is passed a number of consecutive times, the robot returns to translation. In this way, the robot tends to keep the wall within a certain fixed distance from itself as it navigates about the arena.

Using this control scheme, the robot demonstrated the ability to navigate about the previously described arena while avoiding collision with walls. Fig. 7 shows the paths taken by the robot during two such demonstrations. In 7a, it makes CW turns while keeping the wall to its left. In 7b, it turns CCW while keeping the wall to its right. The rotational OF threshold was chosen so that the robot would translate close to parallel to the wall. As shown in Fig. 7, the robot sometimes undershoots and sometimes overshoots the parallel orientation. This is in part due to the differences in distance to the wall from the axis of rotation at each event. Another cause is that the robot's idle wheel has a single axis of rotation that is aligned for the benefit of translation. The increased friction on this wheel during rotation occasionally results in inconstant turning speeds that are sometimes lesser than the average measured speed of 45 °/s. The OF computed during a slower turn is smaller in magnitude and therefore reaches the threshold value sooner.

In addition to navigating the square arena, the robot demonstrated the ability to navigate an environment of

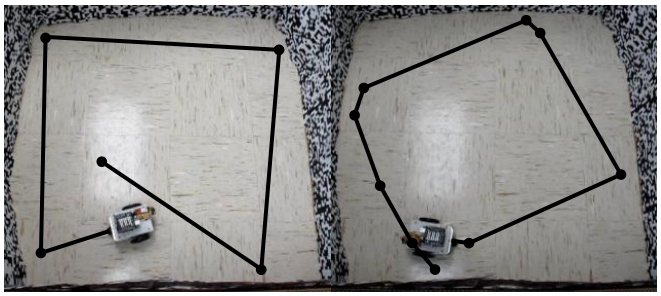

Fig. 7. Robot trajectories during proportional-turn collision avoidance. The robot is shown in its starting position. Black dots indicate points where it rotated. The trajectories were recovered from video.

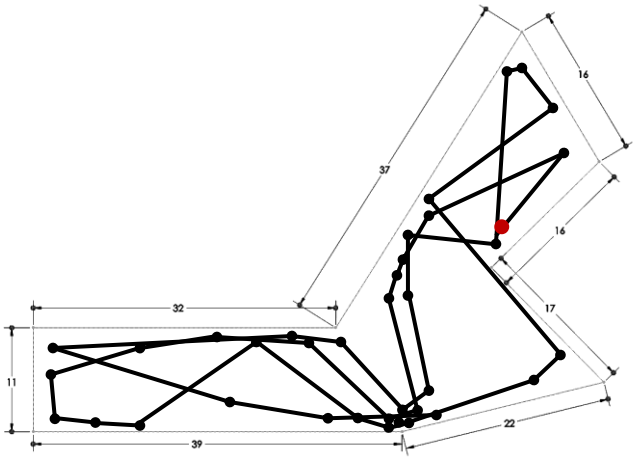

Fig. 8. Robot trajectories during proportional-turn collision avoidance in an environment with arbitrary geometry. Dimensions are in inches. The red dot indicates the robot's starting position. Black dots indicate points where the robot rotated. The trajectories were recovered from video.

unknown and arbitrary geometry (Fig. 8) using the same control strategy. While navigating this environment, the robot attempted to maintain a fixed distance from the wall to its right. Fig. 8 shows the trajectory of the robot as it moved about this environment. Over multiple trials, the robot would occasionally miss the small alcove by passing the opening into it. However, due to variations in its path over time, it eventually found the opening given enough time to explore.

V. CONCLUSION AND FUTURE WORK

Parameters that result in OF with the greatest SNR, while still being computable on the Arduino Uno, were determined. Using OF, a robot was able to navigate a specialized environment while avoiding collision with walls, establishing that the Centeye Stonyman Vision Chip provides sufficient data for small robots to be able to perform certain navigational tasks of interest.

Future work will include mounting the camera on a smaller robotic platform. The TinyTeRP from the Maryland Microrobotics Laboratory has already been chosen and a board is currently being designed to be added to this modular platform. Work with the TinyTeRPs will include studying behaviors of multiple TeRPs that are outfitted with vision sensors. Additionally, data from an inertial measurement unit will be added to the control loop so that the thresholds that trigger rotation and translation events can be adapted to the robot's speed.

VI. ACKNOWLEDGMENTS

The authors would like to thank the Research Experience for Undergraduates program at the Maryland Robotics Center, supported by the National Science Foundation, for the opportunity to conduct this research. Geoffrey Barrows also deserves thanks for developing and providing the vision chip and Rocket Shield and for providing software support on the ArduEye website.

REFERENCES

[1] D. Gage, "Minimum-resource distributed navigation and mapping," in *SPIE Mobile Robots XV*, Boston, MA, 2000.

[2] G. Caprari and R. Siegwart, "Mobile micro-robots ready to use: Alice." IEEE, 2005, pp. 3295–3300.

[3] D. Floreano and L. Keller, "Evolution of adaptive behaviour in robots by means of darwinian selection," PLoS Biology, vol. 8, no. 1, pp. 1-8, 2010.

[4] M. Rubenstein, C. Ahler, and R. Nagpal, "Kilobot: A low cost scalable robot system for collective behaviors," in IEEE International Conference on Robotics and Automation, Minneapolis, MN, May 2012, pp. 3293–3298.

[5] A.P. Sabelhaus, D. Mirsky, L.M. Hill, N.C. Martins, and S. Bergbreiter, "TinyTeRP: A Tiny Terrestrial Robotic Platform with Modular Sensing," in proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2013.

[6] S. Bergbreiter , A. Mehta and K. S. J. Pister "Photobeacon: Design of an optical system for localization and communication in multi-robot systems", *Conf. Robot Commun. Coord.*, 2007.

[7] J.-C. Zufferey and D. Floreano, "Fly-inspired visual steering of an ultralight indoor aircraft," IEEE Trans. Robot., vol. 22, no. 1, pp. 137-146, 2006.

[8] J. Zufferey and D. Floreano, "Toward 30-gram autonomous indoor aircraft: Vision-based obstacle avoidance and altitude control", *Proc. IEEE Int. Conf. Robot. Autom.*, pp.2605 -2610 2005.

[9] A.J. Davison, I. Reid, N. Molton and O. Stasse, Monoslam: Real-Time Single Camera Slam, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052-1067, June 2007.

[10] D. J. Fleet and Y. Weiss. Optical flow estimation. In N. Paragios, Y. Chen, and O. Faugeras, editors, *Handbook of Mathematical Models in Computer Vision*, chapter 15, pp. 239-258. Springer, 2006.

[11] K. Weber, S. Venkatesh, and M. Srinivasan, M. V. Srinivasan and S. Venkatesh, "Insect inspired behaviors for the autonomous control of mobile robots", *From Living Eyes to Seeing Machines*, pp. 226-248, 1997.

[12] D. Coombs , M. Herman , T. Hong and M. Nashman "Real-time obstacle avoidance using central flow divergence and peripheral flow", *ICCV \'95*, pp. 276 -283, 1995.

[13] G. de Croon , K. de Clerq , R. Ruijsink , B. Remes and C. de Wagter "Design, aerodynamics, and vision-based control of the delfly", *Int. J. Micro Air Veh.*, vol. 1, no. 2, pp.71 -97 2009.

[14] M. Srinivasan, "An image-interpolation technique for the computation of optic flow and egomotion," *Biol. Cybern.*, vol. 71, pp. 401-416, 1994.