Inter-vehicle Distance Estimation Using Displaced Stereo Vision

Alfa Budiman, Mathieu Falardeau

University of Ottawa, Faculty of Engineering

Abstract – Vision from an overhead camera mounted on a UAV was combined with vision from a ground based mobile robot. The vision was combined in a manner similar to stereoscopic vision and was used to estimate distance between the ground based mobile robot and a moving target.

Keywords – Stereoscopic vision, Robot Operating System, Unmanned Aerial Vehicle, Unmanned Ground Vehicle

I. INTRODUCTION

Inter-vehicle distance measurement is a very important topic in the domain of self-driving vehicles. It ensures that the vehicle maintains a safe distance with other vehicles and avoid accidental collisions. Different technologies such as laser and radar are available, but they are usually very expensive and difficult to obtain. Alternatively, cameras are much cheaper and accessible options.

Multiple inter-vehicle distance measuring algorithms using cameras were proposed to meet this requirement. One proposed algorithm was to detect edge features of the vehicle and estimate the inter-vehicle distance from the vehicle rear- shadows [1]. Stereo camera was also proposed which determines the distance between the camera and a detected object by detecting the feature points on the images [4] and calculating the object’s image disparity between two cameras with known baseline distance [2]. Over-head cameras are also applied in determining the location of the vehicles [3] which can then determine the distance between the two UGV.

II. problem description

Each of the previously mentioned method have some weaknesses. For stereo vision, this only works if there are a minimum of two cameras on each robot. For companies who want to keep costs low, installing two cameras on each robot might be too expensive. Plus, ground robots must keep visual contact with each other which makes the method pointless if an obstacle is obstructing the cameras. The overhead camera can only measures distance in a 2D plane. This means that the distance measured will be inaccurate if the ground is uneven.

The proposed solution in this paper is to combine both stereo vision and overhead camera. The method is called displaced stereo vision. The concept is to take image captures from cameras attached to a UAV and a UGV and detect the desired object. Afterwards, the method generates two vectors from the camera’s input using non-inverted image projection. Using triangulation, the method calculates the distance between the two UGV.

iII. SySTEM DESCRIPTION

The method was tested using simulated robots within a controlled environment. All robots wee structured using the Robotic Operation System (ROS) which is an open-source robotic middleware suite that is widely used in robotic application. The simulated environment was generated in Gazebo simulator which is an open-source 3D robotics simulator available in ROS. The robots were programmed using the Python programming language, and two libraires were used for image processing and math calculation respectively which were both available in the Python libraries: OpenCV and Numpy. Three robots were used to evaluate the displaced stereo vision method: Hector Quad Rotor (quadcopter figure 1.(a)), Turtlebot 3 Waffle (wafflebot figure 1.(b)) and Turtlebot 3 Burger (burgerbot figure 1.(c)). Both quadcopter and waffleboth acted as the UAV and UGV with cameras and known pose while burgerbot was the target UGV with unknown pose. Additionally, the wafflebot had an integrated lidar sensor which was used to measure the real distance between the two UGV.

A picture containing aircraft

Description automatically generated A picture containing truck, driving, trailer, toy

Description automatically generated A picture containing seat, lawn mower, chair, projector

Description automatically generated

(a) Quadcopter (b) Wafflebot (c) Burgerbot

Fig. 1. Robot 3D model

iv. methodology

The goal is to estimate distances between a target and a UGV. This will be done by combining monocular vision on the UGV’s camera with monocular vision on an overhead camera in a similar manner to [3]. In this case, a quadcopter will act as that overhead camera. This method will occur over 4 steps: object detection, direction calculation, position calculation, and distance calculation.

A. Object Detection

The overhead camera can detect the UGV and the target object. While many methods exist for object detection using vision, for this application, color was used to facilitate ease of detection of the target [8], [9]. The target was a burgerbot and was colored in solid red. The UGV, a wafflebot was colored in solid black.

The overhead camera sees the burgerbot as a group of red pixels. It also sees the wafflebot as a group of black pixels. The red pixel closest to the group of black pixels is the location of the burgerbot form the quadcopter’s perspective. This pixel is annotated with a “+” on figure 1. (a).

A picture containing chart

Description automatically generated A picture containing chart

Description automatically generated

(a) Quadcopter (b) Wafflebot

Fig. 1. Camera Image

The wafflebot camera sees the burgergot also as a group of red pixels. The center of these red pixels is the target’s location from the perspective of the wafflebot. This is shown as the “+” on figure 1.(b). These pixel coordinates, the locations of the “+” on figure 1, are extracted and then used to calculate the direction from the respective observer to the target, as unit vectors.

B. Direction Calculation

After the desired pixel position (and ) are found, the algorithm calculates the unit vector by applying the inverse of the camera projection matrix on the pixel position to determine the 3D ray vector [5] which is shown in figure 3. The simulated camera lenses do not have any distortion, so calibration is not required for this method. The camera’s focal lengths on the x and y axis, and , are equal (1), and because the cameras are monocular, the translation terms and are equal to 0 (2). As such, the new projection matrix was simplified by using matrix (3) where and are the center pixel position of the camera.

Diagram

Description automatically generated

Fig. 3. Camera projection of UGV from real to image plane

|  |  |
| --- | --- |
| , | (1),(2) |
|  | (3) |

The desired vector is calculated by applying the inverse of the projection matrix to the pixel coordinate using equation (4). The unit vector is then calculated by dividing the vector by its magnitude as shown in equation (5).

|  |  |
| --- | --- |
|  | (4) |
|  | (5) |

For the displaced stereo vision to work, the unit vectors must be re-oriented from their respective camera’s reference frame to the world reference frame. Because the quadcopter’s and wafflebot’s pose are known as well as the pose of the camera in reference to the robots, the unit vectors’ orientation was rectified by using quaternion multiplication.

|  |  |
| --- | --- |
|  | (7) |
|  | (8) |
|  | (9) |
|  | (10) |

In ROS, the orientation is represented in quaternions. In equation (7), is the quaternion orientation of the camera in reference to the world frame with vector part and scalar part . To rotate the unit vector in reference to the camera frame, , the method applies the conjugation of two quaternions onto shown in equation (9). This produces the unit vector in reference to the world frame, , which is shown in equation (10).

C. Distance Calculation - Closest Points

A picture containing text, way, light, sidewalk

Description automatically generated

Fig. 3. Wafflebot and Quadcopter Observing Target

Using stereoscopic vision, an object’s location can be calculated if given the direction from two cameras and the locations of these cameras [2]. In figure 1 and equations (1), (2), and are the lines in 3D space from the quadcopter to the target and the wafflebot to the target, respectively.

|  |  |
| --- | --- |
| , | (1),(2) |

and are the location of the quadcopter and wafflebot in reference to the world frame. These locations can be obtained from the onboard sensors on these observers. and are unit vectors from the quadcopter to target and wafflebot to target, respectively. These were obtained from direction calculation:

|  |  |
| --- | --- |
| , | (3),(4) |
| , | (5),(6) |

Ideally, lines and would intersect, but due to the imperfection of the feature matching process in object detection and direction calculation, this will almost never occur. Therefore, the two closest points on non-intersecting lines must be found. The middle between these two points is the location of the target.

The parameters and that yield the closest points are found when the following conditions are satisfied:

|  |  |
| --- | --- |
|  | (7),(8) |

Expanding these dot products yields a system of linear equations that can be expressed in matrix form :

|  |  |
| --- | --- |
|  | (9) |

|  |  |
| --- | --- |
|  | (10) |

Solving this system yields the following values of and:

|  |  |
| --- | --- |
|  | (11) |

Therefore, the target’s location, can be calulcated by substituting the values of and from (11). The distance, , between the wafflebot and the target is the magnitude of the difference between and .

|  |  |
| --- | --- |
|  | (13) |
|  | (14) |

D. Distance Calculation - Sine Law

iv. TESTING AND RESULTS

The distance calculation algorithms were tested by having the target drive through four waypoints as per figure 3. Two tests were conducted; test 1 with the wafflebot as a stationary observer and test 2 with the wafflebot following the burgerbot at a preset distance.

A picture containing chart

Description automatically generated

Fig. 4. Test Drive Waypoints

The results of test 1 are on figure 5. The results of test 2 are on figure 6. Both figures 5 and 6 plot the calculated distance from visual data, using methodology described in section IV - C and D, as well as the measured distance from the wafflebot’s onboard lidar.

For test drive 2 a simple proportional controller was used to control the wafflebot’s position and orientation. This constituted a simplified form of visual servoing [10]. The wafflebot’s linear speed, was proportional to the distance between it and the target. Where is a control gain and was calculated from (14).

|  |  |
| --- | --- |
|  | (15) |

The wafflebot’s rotational speed was proportional to the angle between it and the target. This was calculated as the angle between the vector of the wafflebot’s direction , and the unit vector in from the wafflebot to the target :

|  |  |
| --- | --- |
|  | (16) |

is obtained from direction calculation and is a control gain. is known from the sensors on the wafflebot.

Chart, line chart

Description automatically generated

Fig. 5. Distance Measurements for Stationary Trial

Chart, histogram

Description automatically generated

Fig. 6. Distance Measurements for Leader - Follower Trial

V. DISCUSSION AND CONCLUSION

From figures 5 and 6, it can be observed that the calculated distances (obtained from section IV - methodology) closely tracked the lidar distance in the stationary trial. However for the leader follower trial, the calculated distances were not entirely consistent with the measured lidar distance.

During the leader-follower trial however, the error, which is (difference between the calculated distances and the lidar distance) was less consistent. The error increased as the distance between the wafflebot and burgerbot increased, and then the error decreased as the wafflebot’s motion was reduced.

This error was caused from the imperfection of the feature matching process. As the wafflebot moved its pixel location on the quadcopter camera changed, and that change may not be completely consistent with its motion. This change combined with a similar change in the burgerbot’s pixel location resulted in a noisier signal. However, it must also be noted that the overall distances are relatively small to the sizes of the objects. If the overall distances were larger compared to the objects, then the (relative) error between the calculated distances and lidar distance would be significantly smaller.

Overall, the vision from both the quadcopter and the wafflebot was combined into a distance estimation solution. This solution can be applied to provide additional information for coordination of multiple robots. For example, a group of robots may be required to move to a location while maintaining formation, this form of platooning is possible using only monocular vision [6]. However, monocular vision for distance estimation requires additional information on the dimensions of objects in its environment.

This maybe an issue when multiple different types of robots have to cooperate with each other or with human operated elements such as in [7]. It may not be feasible to rely solely on monocular vision. Furthermore, while it maybe possible to add additional sensors to these platforms, it still relies on these platforms maintaining line of sight to the targets / objects of interest.

This overhead view can track objects even if UGVs lose sight of the targets. This additional information can then be combined with previously collected data and then a path to the target can be generated, enabling the UGV to re-acquire it. At the same time, relying solely on an overhead camera has limitations, namely, it requires additional information on the nature of the ground (changes in elevation, slope, camera height). Combining an overhead monocular camera with UGV mounted monocular cameras provides additional information to create a complete picture of the situation.

REFERENCES

1. G. Kim and J. -S. Cho, "Vision-based vehicle detection and inter-vehicle distance estimation," *2012 12th International Conference on Control, Automation and Systems*, 2012, pp. 625-629.
2. Abdelmoghit Zaarane, Ibtissam Slimani, et all, “Distance measurement system for autonomous vehicles using stereo camera,” *Array Volume 5, 2020, 100016, ISSN 2590-0056*
3. Z. Ziaei, R. Oftadeh and J. Mattila, "Vision-based path coordination for multiple mobile robots with four steering wheels using an overhead camera," *2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 2015, pp. 261-268, doi: 10.1109/AIM.2015.7222542.
4. Y. Shima, "Inter-vehicle distance detection based on keypoint matching for stereo images," *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, 2017, pp. 1-6, doi: 10.1109/CISP-BMEI.2017.8302064.
5. M. T. Bui, R. Doskocil and V. Krivanek, "Distance and angle measurement using monocular vision," *2018 18th International Conference on Mechatronics - Mechatronika (ME)*, 2018, pp. 1-6.
6. P. Avanzini, B. Thuilot and P. Martinet, "Accurate platoon control of urban vehicles, based solely on monocular vision," *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 6077-6082, doi: 10.1109/IROS.2010.5650018.
7. C. J. R. McCook and J. M. Esposito, "Flocking for Heterogeneous Robot Swarms: A Military Convoy Scenario," *2007 Thirty-Ninth Southeastern Symposium on System Theory*, 2007, pp. 26-31, doi: 10.1109/SSST.2007.352311.
8. J. Li, J. Wang and J. Mao, "Color moving object detection method based on automatic color clustering," Proceedings of the 33rd Chinese Control Conference, 2014, pp. 7232-7235, doi: 10.1109/ChiCC.2014.6896196.
9. A. N. Fitriana, K. Mutijarsa and W. Adiprawita, "Color-based segmentation and feature detection for ball and goal post on mobile soccer robot game field," 2016 International Conference on Information Technology Systems and Innovation (ICITSI), 2016, pp. 1-4, doi: 10.1109/ICITSI.2016.7858232.

[10] H. Mekki and M. Letaief, "Path planning for 3D visual servoing: For a wheeled mobile robot," 2013 International Conference on Individual and Collective Behaviors in Robotics (ICBR), 2013, pp. 86-91, doi: 10.1109/ICBR.2013.6729262..