

Developing and Modelling of Satellite Docking Algorithm

Y. Erdem Aras M. Numan Uyar, Mahmut C. Soydan,
M. Selahaddin Harmankaya, Furkan Alan
Department of Electrical and Electronics Engineering
Bilkent University
Ankara, Turkey 06800
Email: {yunus.aras, mustafa. mahmut.soydan,
muhammed.harmankaya, furkan.alan}@ug.bilkent.edu.tr

Burak Akbulut
Space Systems,
Turkish Aerospace Industries (TAI)
Ankara, Turkey 06980
Email: bakbulut@tai.com.tr

Abstract—In this study, a stereovision sensor system hardware and an algorithm are developed to be utilized in autonomous satellite rendezvous applications. A two dimensional representative environment is also being developed consisting of omni wheel robots to test breadboard model of the sensor system in a similar proximity operation scenario. The sensor hardware is designed according to predefined requirements keeping the limitations of the satellites. The sensor is able to estimate position, orientation, linear and angular velocity of a target object whose shape and sizes are known a priori. The detection system relies on specific reference markers and extracted BRISK feature points. Both of stereo and monocular vision approaches are used to detect object and estimate its distance, followed by reverse rigid body transformation to estimate the target's 3D location and orientation. Time difference between two subsequent frames is used for estimating linear and angular velocities. Additionally, a path- planning algorithm is developed to approach target object in an efficient way.

Index Terms—autonomous; rendezvous; stereo vision; monocular; spacecraft docking

I. INTRODUCTION

Different types of sensor systems are available for autonomous satellite rendezvous applications. From a CubeSat application standpoint, commonly used sensors in satellite rendezvous applications have some disadvantages. For example, Jena LIDAR sensor [1] and Advanced Video Guidance Sensor (AVGS) [2] have high mass and high power consumption for pico satellite applications. SpaceX Dragon 3D Flash LIDAR Space Camera [3] is lighter and smaller compared to Jena LIDAR and AVGS; however, yet it is also not suitable for CubeSat applications as presented in Table???. In this regard; a lightweight, low power consumer, compact and low cost stereo vision camera sensor system is proposed.

The objective of this study is to design and set up a visual guidance system that could be utilized in au-

TABLE I
AVAILABLE CUBESAT APPLICATIONS

	Jena LIDAR Sensor	AVGS	SpaceX-DragonFive
Weight	8 kg	9 kg	3 kg
Size	28.6 cm x 31 cm x 19.5 cm	17.78 cm x 25.4 cm x 30.48 cm	11.2 cm x 13.2 cm x 11.9 cm
Power Consumption	30 W	35 W	35 W
Technology	LIDAR	Video-based laser-illuminated sensor	LIDAR
Range	3000m	300m	1500m
Mission	Actively used	since 2007	since 2012

tonomous satellite rendezvous applications. The sensor system uses both of stereo vision camera and monocular camera to estimate position, orientation, linear velocity and angular velocity of the target object with respect to the satellite having the sensor. When the target is far away, the stereo vision camera and disparity map algorithm is used to detect object and its distance. After the distance is decreased to a predefined value, monocular camera and a novel algorithm is used to detect target and estimate its position and orientation. Also, another novel approach is to utilize an IR camera to deal with low and inadequate lighting conditions in orbit.

The specifications of the proposed stereo vision system are given in Table II. It is a compact, lightweight, high performance and low power consumer system. The visual

TABLE II
SPECIFICATIONS OF SENSOR SYSTEM

Properties	Requirements
Weight	<500g
Power consumption	<5W
Size	5cm × 10cm × 10cm i.e. Half Unit CubSat
Output Rate	>2 Hz
Output	Position, Orientation, Linear and Angular Velocity

guidance system developed in this study was tested in simulation environment by using Omni wheel robots representing the both chaser and target satellites and satellite mockup. The tests were carried out while target object is stationary, and has angular velocity. The chaser robot having sensor system on detects the target mockup on the other Omni wheel robot, and approaches to it in an efficient way by following a path-planning algorithm.

II. HARDWARE SELECTION

Essentially requirements for the sensor design and budget of the project have influences on hardware preferences. The dimension and power consumption requirements of the project confined embedded system options to FPGA and small embedded systems. FPGA based solutions meet the requirements of the sensor system; however, they are not the most desired platforms because VHDL is a complex programming language for image processing applications. Also, OpenCV library contains readily many image processing functions. Therefore, C++ programming language is the most appropriate option for coding language selection. This situation brings about considering embedded systems that supports C++. Raspberry Pi 3 Model B platform was the initial preference due to common usage and budget related issues among embedded systems. Since it is aimed to implement algorithm in C++ language, it is easy to change the platform in the future. Therefore, it is planned that if Raspberry Pi platform exhibit shortcomings, it can be easily switched to higher performance and higher cost solutions such as NVidia's Jetson chips.

For camera selection, two limiting factors exist. First, the algorithm execution time, which must be lower than 500 msec (i.e. > 2 Hz), increases with high resolution images due to increase in data to process. Secondly, budget restrictions also limit the options. By considering these confronts, the high-end cameras are not preferred, and Logitech C270 model webcam was selected. For IR camera selection, Pi NoIR Camera V2 developed for Raspberry Pi by the Raspberry foundation was chosen because it is budget friendly, easy to obtain and have ready interface with Raspberry Pi.

III. ALGORITHM DESIGN

For target detection process, disparity map algorithm and monocular vision approach were used. Disparity map algorithm enables to straightforwardly obtain the binary map of the sensor vision and to detect the target without searching its feature points even from long distances; however, its processing rate 0.5 Hz. Monocular vision approach processes much faster than the disparity map, around 2-5 Hz, and not only detects the target, but also estimates its position and orientation. Detecting and tracking the target is important when satellites are far away from each other, and accuracy of the estimations and frame rate of the cameras are important when satellites are close. Therefore, disparity map algorithm is used when satellites are far away and monocular vision is used when they are close. The applied algorithm is changed after a certain predefined value, which is preferred as 2 m in this study's tests, but can be changed easily.

A. DISPARITY MAP ALGORITHM

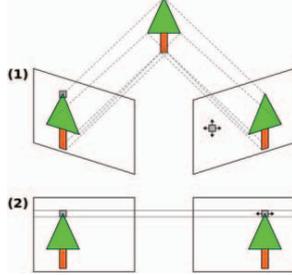
Disparity map is a common subject in stereo vision applications [5]. In stereo vision systems, two cameras are placed with a specified base length and their views are combined on the same scene. The relative depth map of the scene can be produced by comparing these two images in disparity map format. The values in this map are inversely related with the real depth values [6]. Since the 3D position of the target is desired to be obtained from the stereo vision, mentioned method is thought to be useful in such field. The main approach in this method will be the following:

- Obtain stereo images from stereo vision camera
- Rectify the images so that images are co-planar
- Obtain disparity map
- Reconstruct 3D scene
- Obtain 3D position of the target

Image rectification is used to combine images on a common plane as they will project onto other one. It simplifies the problem of finding matching points between images [7]. The Fig. 1 illustrates how image rectification simplifies the search space in stereo correlation matching.

In order to be able to apply rectification, some parameters of the stereo-vision need to be known. For that purpose, at first stage the stereo-vision cameras should be calibrated in offline. 30 pairs of stereo-vision images are taken in different position and orientation by using known size reference material, such as checkerboard. In the calibration process, distortion coefficients and characteristics of the cameras are determined.

Fig. 1. The search space before (1) and after (2) rectification



After applying the rectification, the images are converted to co-planar images, which means each corresponding pixels lie in the same line and it makes easier to perform disparity map, since the search of each pair of pixel becomes easier. The images before and after rectification are shown in Fig. 2. Subsequently, using rectified images and the found calibration parameters the disparity map of the environment is obtained as shown in Fig. 2. Each object is coloured in different grey shades according to its distance to stereo-vision camera. Thus, distance to the target can be estimated using disparity map. Since noise exists in cameras, Gaussian filtering is applied to the disparity map as displayed in Fig. 3. Also, different filtering methods can be used to filter the scene and to easily find the searched object. As a final step, 3D scene of the images is reconstructed by using deepness info from the disparity map, and the pixel of the target is found from disparity map. 3D position of the center of the target is found from 3D reconstructed scene and the distance is calculated.

B. MONOCULAR VISION

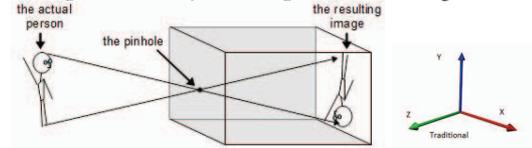
In this method, ?Pose Estimation? algorithm and OpenCV libraries are used to detect the object. Pose Estimation is an algorithm that is commonly used for detection of planar surfaces since its detection accuracy is sufficiently high. This algorithm uses feature point matching algorithms to detect the target object. Some different feature matching algorithms such as ORB, FAST and BRISK are tested and BRISK is preferred because number of found feature points and process time of this algorithm fit best to this sensor.

After target object is detected, the position and orientation of the target object is calculated by using a novel algorithm proposed by this work. This algorithm bases on knowing the size and shape of the target object, and technical specifications of the camera a priori. The position and orientation of the target object can easily be

Fig. 2. Raw images taken from left and right cameras (top), red-cyan composite view of the stereo images before (middle-left) and after (middle-right) rectification, disparity map (bottom-left) and Gaussian filtered disparity map (bottom-right)



Fig. 3. The Proportion Logic Behind the Algorithm



calculated with basic proportion rules. The mathematics behind the algorithm is derived from the pinhole cameras.

As it is seen from Fig. 3, the direction of any pixel can be calculated from the image. That means, any pixels x (width) and y (height) coordinates can be written in terms of z (distance) coordinates of the pixel:

$$\frac{cx_1}{cz_1} = \frac{rx_1}{rz_1} \Rightarrow rx_1 = \frac{cx_1 \times rz_1}{cz_1} \quad (1)$$

$$\frac{cy_1}{cz_1} = \frac{ry_1}{rz_1} \Rightarrow ry_1 = \frac{cy_1 \times rz_1}{cz_1} \quad (2)$$

where ;

- **cx, cy and cz:** coordinates of images pixels according to the pinhole,
- **rx, ry and rz:** coordinates of real points according to the pinhole.

In these equations, the only unknowns are z coordinates of real points. If 3 point is chosen, 3 different equations are obtained to find exact location of the points. For these

equations are derived from the distance formula between the points:

$$L_{12}^2 = (rx_1 - rx_2)^2 + (ry_1 - ry_2)^2 + (rz_1 - rz_2)^2 \quad (3)$$

$$L_{13}^2 = (rx_1 - rx_3)^2 + (ry_1 - ry_3)^2 + (rz_1 - rz_3)^2 \quad (4)$$

$$L_{23}^2 = (rx_2 - rx_3)^2 + (ry_2 - ry_3)^2 + (rz_2 - rz_3)^2 \quad (5)$$

These second degree 3 equations are needed to be solved to find the real locations. However, solution may be more than one since they are second degree. By increasing the number of points, a unique solution can be found; whereas, still have to deal with second degree equations, which slow down the algorithm. Instead of those formulas, a specific case can be solved more easily. If those 3 points are linear and let's say 3rd point is in middle of the other two points, it is found that:

$$rx_2 = \frac{rx_1 + rx_2}{2} \quad (6)$$

$$ry_2 = \frac{ry_1 + ry_2}{2} \quad (7)$$

$$rz_2 = \frac{rz_1 + rz_2}{2} \quad (8)$$

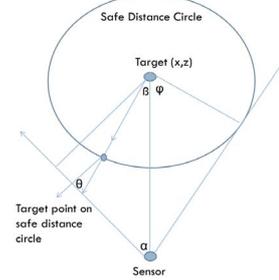
Thus, 3 different equations are obtained. When calculations are done, unique solutions for each point are found. The real locations of the chosen points can be calculated as well as the real position of the target object.

After finding the location of the rigid body, those 3 points are assumed as a vector to find the orientation of the target object. However, to calculate its orientation, 2 vectors which are not parallel are needed. Therefore, another 3 points are needed. After finding 2 unit vectors of the rigid body, rigid body transformation is applied and the orientation is found. As a result, with total of 5 points, the location and orientation of a rigid body can be calculated with a single camera.

C. Path Planing Algorithm

This algorithm outputs an efficient path for chaser robot to approach to the target robot. The model of the simulation environment is shown in Fig. 4. Firstly, chaser robot moves linearly till the safe distance circle whose center is target robot and radius is 60 cm. 'x' and 'z' are the relative coordinates of the target object to the sensor system (sensor is located at the origin and looking view is the y axis of the coordinate system). α is the angle between normal vector of the sensor and the line that connects robots. β is the angle between normal vector of the target object and the line that connects robots. θ is the angle between normal vectors of the sensor system and

Fig. 4. The model of the simulation environment



target object, which is also the value calculated by the sensor system. φ is the critical angle for sensor system to be able to see docking port of the target object.

Firstly, if $\beta < \theta$, chaser robot moves linearly to the just opposite of the target object's docking port. If not, it is impossible to directly go to the opposite of the docking port. Therefore, it moves by assuming $\beta = \theta$. To make these linear movement, the velocity in x direction, the velocity in z direction and angular velocity are chosen as proportional with $(x - SafeDistance \times \sin(\theta))$, $(z - SafeDistance \times \cos(\theta))$, and θ , respectively. Secondly, chaser robot makes a circular movement to dock to the target robot. At this stage, the velocity in x direction, the velocity in z direction and angular velocity are selected as proportional with $(SafeDistance \times \theta)$, z, and θ , respectively.

IV. TEST ENVIRONMENT

A. Omni Wheel Robot Hardware

A two dimensional representative environment of the space is being developed using omni wheel robots which are 15001 3WD 48 mm Omni Wheel Robot Platform Chassis by Nexus Robot. These robots have three 12 V DC motors which drive omni wheels with diameters of 48 mm. Omni wheel can either roll like a normal wheel or roll laterally using the wheels along its circumference; thus, it can provide the ability to move any direction on a plane without the need for a traditional swivel mount. These wheels enable the robot to move forward/backwards and rotation without changing its orientation as well as planar rotation. The hardware of omni wheel robot is shown in Fig. 5.

B. Omni Wheel Robot Dynamics & Kinematics

A torque is applied on the omni wheels according to its input voltage. It is found out that there is a linear correlation between input voltage applied on the omni wheels and angular velocity of the omni wheels. Lets say forces applied on the wheels are

Fig. 5. Hardware of Omniwheel Robot

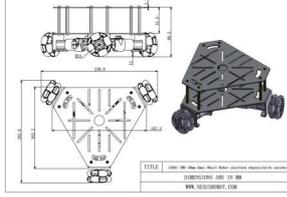
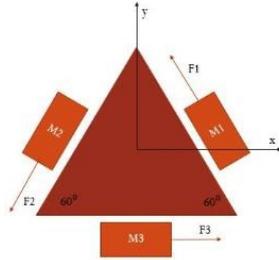


Fig. 6. Free-body Diagram of the Omniwheel Robot



$F_w = [F_1 \ F_2 \ F_3]^T$, velocities of the wheels are $\dot{X}_w = [r \times q\dot{L}_1 \ r \times q\dot{L}_2 \ r \times q\dot{L}_3]^T$ where r is the radius of the omni wheel and $q\dot{L}_i$ s are angular velocities of the wheels. The force on the robot is $F_m = [F_{m_x} \ F_{m_y} \ T_z]^T$, and velocity of the robot in its moving frame is $X_m = [x_m \ y_m \ \varphi_m]^T$. The free-body diagram of the omni wheel robot is shown in Fig. 6.

When forces are applied on the wheels, the total force on the robot can be determined easily using the shape of the omni wheel robot. The relation between forces on wheels and total force on the robot is:

$$F_m = \begin{bmatrix} 1 & \cos(30) & \cos(150) \\ 1 & \cos(240) & \cos(120) \\ L & L & L \end{bmatrix} F_w = B F_w \quad (9)$$

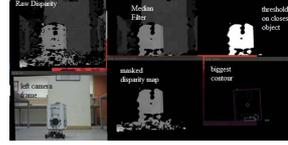
Here L is the distance between center of the robot and the wheel. Then, since the friction is ignored for omni wheel robot, the work done by the wheels must equal to the work done by the robot by using the principle of virtual work. This brings about $F_m \dot{X}_m = F_w \dot{X}_w = F_w r (q\dot{L})$. By combining equation (1) and (2), the relation between velocity of the robot in its moving frame and angular velocities of the wheels becomes that

$$\dot{X}_m = (B^T)^{-1} r q\dot{L} \quad (10)$$

V. TEST RESULTS

The algorithms were tested with a satellite mock up used by NASA in the past. The algorithms are coded in C++ and tested with Raspberry Pi 3 Model B.

Fig. 7. Original Scene, Disparity Map and Filtered Scene of the Environment



A. Disparity Map Algorithm Test Results

Common disparity map algorithm in literature only produces the disparity map of the environment. After that point, some filters are applied to the disparity map of the environment in this study. Disparity map is filtered on the base of the assumption that closest object to sensor system is target.

To apply these filters, a threshold value is obtained by getting closest point's colour value. Median filter and found threshold value are used to get reasonable closest objects depth value inside disparity map. The threshold is used for binary masking where the values higher than given threshold is 1 and lower ones are 0. 1 represents white and 0 represents black. Then, the filtered disparity map is masked according to the most close distance value. Thus, the biggest contour and its center can be found. Afterwards, color values of binary mask and disparity map is intersected and the scene is reconstructed. As a result, the object can be successfully detected. The filtered images of the environment is shown in Fig. 7

The processing rate of this algorithm is about 0.5 Hz with Raspberry Pi. It is not preferred in close distances since its processing rate is quite low. Using two cameras and filtering algorithms are some reasons of long processes. However, the algorithms rate reaches 5 Hz with a standard notebook computer. That means its speed can be increased up to 5 Hz by using better embedded systems.

The distance of the target object can also be found with this algorithm with 30 cm error. Error increases as distance between object and sensor system increased. This algorithm is developed to use when distance is more than 2 meters; therefore, this error rate is sufficiently good for purpose.

B. Monocular Vision System Test Results

The position calculations of the sensor are tested with Sharp GP2Y0A710K0F distance sensor. The tests were undergone when object is moving on a linear path forward and back with constant speed at different positions and orientations. Some measurement results for z position of the object are shown below in Fig. 8:

The z position error of the sensor is approximately less than 2 cm. Its accuracy is better when distance is

Fig. 8. The z Position Measurements of the Sensor

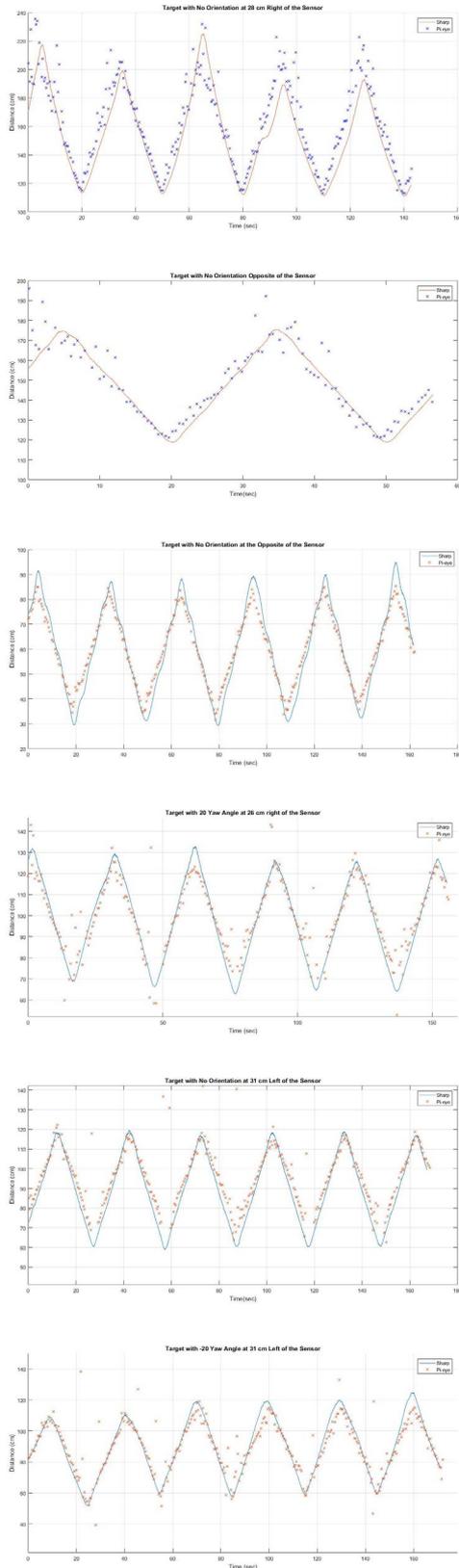
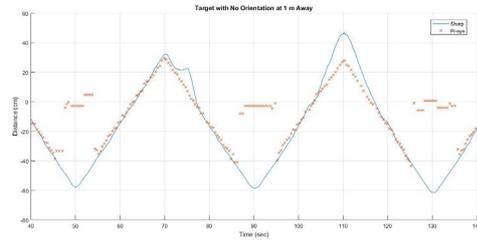


Fig. 9. The x Position Measurements of the Sensor



less than 1 meter and sensor has a good looking angle to the objects one face. These results were obtained when camera has resolution of 640x480. The accuracy of the sensor increases with higher resolutions because object can be detected better and points are selected more accurately. The x position measurements of the sensor are given below in Fig.9 :

These results were measured when z coordinate of the target is fixed to 1 m and y is also fixed. The differently measured parts are that when object gets out of the field of view of the sensor. That's way the measurements are different. The error around second 110 is most probably a measurement error because the object was doing the same movement, and sensor recorded almost the same measurements.

The orientation calculation results were tested with Arduino 9-Axes Motion Shield. The tests were undergone when the objects position is fixed at 70 cm and 140 cm. Results are shown Fig. 10. While yaw angle calculation results fit well with gyroscope results at 70 cm, error increases when distance is 140 cm. That's why 5 points to calculate orientation cannot be detected exactly as calculated.

The processing rate of the algorithm changes between 2-5 Hz. It mostly depends on the embedded system. If a more powerful system is used, it increases. It reaches to 20-25 Hz rate with a standard notebook computer. Measurements of process rate of the sensor is given in Fig. 11

C. IR Camera Measurements

When target object is excessively illuminated, RGB cameras struggle to detect the target; however, IR camera easily detects target without any drop in processing rate. Additionally, when sunlight does not exist around the target object, target is illuminated by only electromagnetic waves in the space. By using the phenomena of All of the planets and moons in our solar system emits strongly in the infrared [8], space environment was simulated with IR leds and capability of the IR camera was tested. As

Fig. 10. The Orientation Measurements of the Sensor

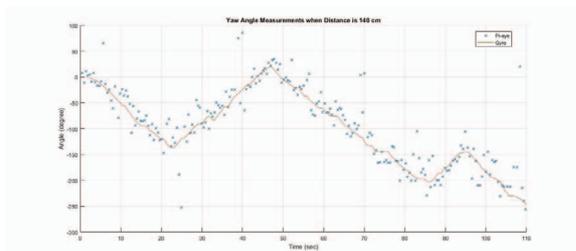
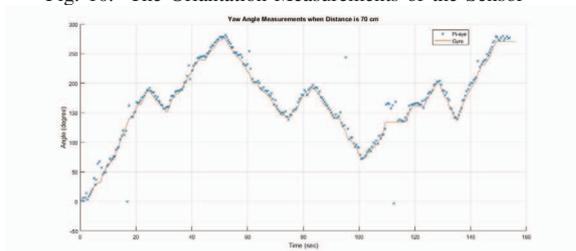
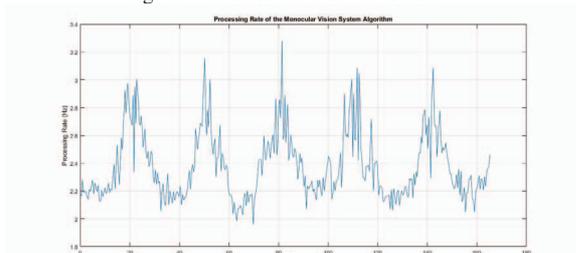


Fig. 11. Process Rate of the Sensor

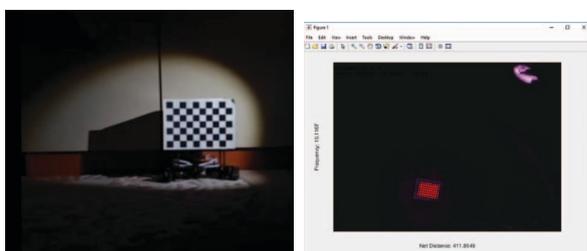


as a result, IR camera detects the target without processing rate drop even when there is no sun light in the environment. The measurement results are shown in Fig. 12.

VI. CONCLUSION

In this study, a compact, light, low power consumer and high performance visual guidance system for satellite rendezvous applications is developed and presented. The

Fig. 12. IR Camera Measurements with IR LEDs in the Dark



developed sensor system can successfully estimate the position, orientation, linear and angular velocities of the target regardless of lighting conditions. The system uses feature point extraction to detect the target, uses stereo and monocular vision approaches and reverse rigid body transformation to calculate position and orientation. Additionally, a two dimensional representative space environment was developed using omni wheel robots to test breadboard model of the designed sensor system and a path-planning algorithm was developed to approach to the target in an efficient way.

VII. ACKNOWLEDGMENT

The authors would like to thank TAI for their cooperation, Prof. Dr. Orhan Arkan and Prof. Dr. Omer Morgul for their priceless mentorship through this study.

REFERENCES

- [1] jenaoptronik, "RVSTM 3000;". [Online]. Available: http://www.jenaoptronik.de/en/aocs/rvs.html?file=tl_files/pdf/Data%20Sheet%20RVS%203000.pdf. Accessed: Feb. 16, 2017.
- [2] R. T. Howard and T. C. Bryan, "The Next Generation Advanced Video Guidance Sensor: Flight Heritage and Current Development;". [Online]. Available: <https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20090033117.pdf>. Accessed: Feb. 16, 2017.
- [3] A. S. Concepts, "Dragoneye;". 2015. [Online]. Available: <http://www.advancedscientificconcepts.com/products/older-products/dragoneye.html>. Accessed: Feb. 16, 2017.
- [4] T. D. Cole, "NEAR Laser Rangefinder: A Tool for the Mapping and Topologic Study of Asteroid 433 Eros;". [Online]. Available: <http://www.jhuapl.edu/techdigest/TD/td1902/cole.pdf>. Accessed: Feb. 16, 2017.
- [5] R. A. Hamzah and H. Ibrahim, "Literature survey on stereo vision disparity map Algorithms;". Journal of Sensors, vol. 2016, pp. 123, 2016.
- [6] N. Lazaros, G. C. Sirakoulis, and A. Gasteratos, "Review of stereo vision Algorithms: From software to hardware;". International Journal of Optomechatronics, vol. 2, no. 4, pp. 435462, Nov. 2008.
- [7] G. Gerig, "Image Rectification (Stereo);". [Online]. Available: <http://www.sci.utah.edu/~gerig/CS6320-S2013/Materials/CS6320-CV-F2012-Rectification.pdf>. Accessed: Feb. 16, 2017.
- [8] "THE INFRARED UNIVERSE - our solar system;". [Online]. Available: <http://www.ipac.caltech.edu/outreach/Edu/ssys.html>. Accessed: Feb. 10, 2017.