Journal of Computer Sciences and Applications, 2018, Vol. 6, No. 1, 17-22
Available online at http://pubs.sciepub.com/jcsa/6/1/2

DOI:10.12691/jcsa-6-1-2

Computational Vision for Automatic Tracking and Objective Estimation of Mobile Robot Trajectory

Sangho Park*

Department of Computer, Electronics and Graphics Technology, Central Connecticut State University, USA
*Corresponding author: spark@ccsu.edu

Abstract Automatic tracking and evaluation of moving-object trajectories is critical in many applications such as performance estimation of mobile robot navigation. Mobile robot is an effective platform for stimulating student motivation at K-12 institutions as well as a good tool for rigorous engineering practices in colleges, universities, and graduate schools. Developing new mobile robot platforms and algorithms requires objective estimation of navigation performance in a quantitative manner. Conventional methods to estimate mobile robot navigation typically rely on manual usage of chronometer to measure the time spent for the completion of a given task or counting the success rate on the task. This paper proposes an alternative; a multi-camera vision system that can automatically track the movement of mobile robot and estimate it in terms of physics-based profiles: position, velocity, and acceleration of the robot in the trajectory with respect to a user-defined world-coordinate system. The proposed vision system runs two synchronized cameras to simultaneously capture and track the movement of the robot at 30 frames per second. The system runs a homography-based projection algorithm that converts the view-dependent appearance of the robot in the camera images to a view-independent orthographic projection mapped on the registered world coordinate system. This enables the human evaluator to view and estimate the robot navigation from a virtual top-down view embedded with the physics-based profiles regardless of the actual cameras' viewing positions. The proposed system can also be used for other domains including highway traffic monitoring and intelligent video surveillance.

Keywords: *computational vision, object tracking, trajectory estimation, robot navigation, multiple view geometry*

Cite This Article: Sangho Park, "Computational Vision for Automatic Tracking and Objective Estimation of Mobile Robot Trajectory." *American Journal of Systems and Software*, vol. 6, no. 1 (2018): 17-22. doi: 10.12691/jcsa-6-1-2.

1. Introduction

Mobile robot development and competition has become very popular in STEM (Science, Technology, Engineering and Mathematics, [1]) education. Mobile robot is an effective platform for stimulating student motivation at K-12 institutions as well as a good tool for rigorous engineering practices in colleges, universities, and graduate schools. In robot design and development at all levels of institution, it is import to objectively measure and estimate the mobile robot's navigation performance. However, the usual practice in the performance evaluation of mobile robot navigation is typically based on human evaluator's manual intervention using a chronometer to measure the time of completion in a given task or accuracy counting of pass / fail on the task. The manual intervention is error-prone and can be biased as well. We need a tool that reliably and objectively evaluates the performance in an automated manner. It is also desirable to develop an unobtrusive method that does not require the attachment of any sensors, transponders, or beacons to the mobile robot since such attachment will change the weight of the robot not alone the complicacy of the installation and management of such attachment. A computer vision-based object detection and tracking approach is a promising solution in this regard. Review of general research on vision-based object detection and tracking can be found in [2,3,4].

This paper presents a vision-based evaluation testbed for mobile robot navigation by using multiple cameras. The proposed system automatically records the movement of the robots and objectively estimates their navigation performance. Unlike the methods for robot's self-localization using heterogeneous sensors and robot models [5,6], the proposed system provides a purely vision-based testbed that evaluates the navigation performance in terms of the physics-based profiles: position, velocity, and acceleration of robot over time with respect to a given world-coordinate system.

2. Methodology

Our methodology is based on multiple-view geometry [7] in computational vision. The method starts by modeling the process of image formation when a scene is viewed through a camera. A pinhole camera model is adopted as shown in Figure 1. Pinhole camera model [8] assumes that exactly one ray from each point in the scene passes through the pinhole lens and hits the image plane opposite to the scene, forming the inverted image. For convenience we use the virtual image in front of the

pinhole, forming the new image plane as shown in Figure 2. The mathematical description of the imaging process in Figure 2 denotes the world scene coordinates with uppercase roman letters {X, Y, Z} and image coordinates with lowercase roman letters {x, y, z}. Note that the vectors pointing from camera center C to the world coordinate point and the corresponding image coordinate point are denoted by boldface symbols, such as $\boldsymbol{X}$ and $\boldsymbol{x}$, respectively. The camera's imaging process is to map the point $\boldsymbol{X}$ denoted by coordinate value (X,Y,Z) in the 3D space to the point $\boldsymbol{x}$ denoted by coordinate value (x,y) on the 2D image plane. This process is mathematically modeled in equation (1) as follows (,where the superscript t means column vector notation.)

$$(\mathrm{X},\mathrm{Y},\mathrm{Z})^{\mathrm{t}} \xrightarrow{\text{Projection}} (\mathrm{x},\mathrm{y})^{\mathrm{t}} \tag{1}$$

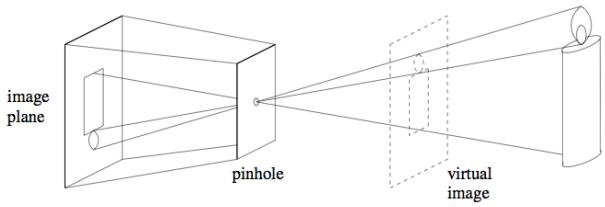

Figure 1. Pinhole camera model. From Forsyth & Ponce, 2003

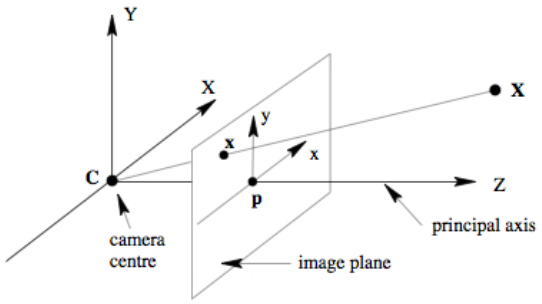

Figure 2. Image projection in a projective camera

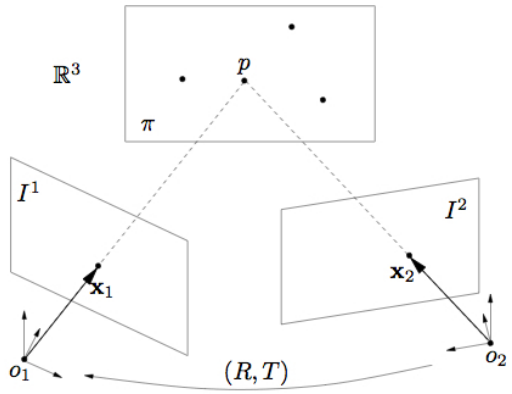

Figure 3. Two views of points on a plane $\pi \subset \mathbb{R}^3$. From Ma et al., 2001

As shown in Figure 3, if we use two cameras, each of the camera forms image plane I^1 and I^2 with the corresponding camera origins O_1 and O_2, respectively [9]. The second camera's relative configuration with respect to the first camera can be defined in terms of the relative rotation R and translation T. Note that the same point p in the world scene π appears very different in the two image planes as vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, respectively, due to perspective distortion effect. Overall, the camera imaging process is mathematically modeled as the mapping of a viewed object from the 3D-world scene coordinate (X,Y,Z) to the 2D image plane coordinate (x,y). This kind of mapping from higher dimension to lower dimension results in inevitable loss of information during the downgrading transformation from the higher to lower dimension. For example, depth information of a scene is lost in 2D photo.

Our ultimate goal in computational vision is to recover the lost information of the viewed object by the inverse mapping from the 2D image coordinate (x,y) $\in \mathbb{R}^2$ on the image planes I^1 and I^2 to the world scene coordinate (X,Y,Z) $\in \mathbb{R}^3$ in the world scene π. This inverse mapping will enable the accurate estimation of the world scene given only the 2D image data. For this purpose, we need at least two camera views in order to resolve the ambiguity caused by the information loss during the initial downgrade transformation. To resolve the ambiguity, we first conduct the camera calibration that establishes the camera configuration (R,T) of Figure 3, as expressed in equation (2).

$$\boldsymbol{x_2} = R\boldsymbol{x_1} + T. \tag{2}$$

A perspective projection for planar homography [7] is used to do the inverse mapping from image- to world-coordinate system. A homography matrix H maps corresponding points between image coordinate systems. If we denote H_m^n as the homography from arbitrary view m to n, we can register multiple arbitrary camera views by the series of concatenated homographies as shown in equation (3).

$$H_m^n = H_{n+1}^n\, H_{n+2}^{n+1} \cdots H_{m-1}^{m-2}\, H_m^{m-1}. \tag{3}$$

The 4-point algorithm [10] is used to compute the homography matrix H. The four points are selected from certain image corners or user-specified markers on the ground plane Q.

Using the homography matrix H, we can write the transformation of points in 3D from camera i to camera j as in equation (4):

$$X_j = HX_i, X_i, X_j \in \mathbb{R}^3. \tag{4}$$

The j-th camera can even be a virtual camera viewing the scene from top down. A point in view-1, $\boldsymbol{x}_1$, and a point in view-2, $\boldsymbol{x}_2$, of the same 3D point q are mapped to x_1^v and x_2^v on the common virtual view of the ground plane by individual planar homography matrices H_1^v and H_2^v, respectively in equations (5) and (6), as follows:

$$x_1^v = H_1^v\, x_1 \tag{5}$$

$$x_2^v = H_2^v\, x_2 \tag{6}$$

The two versions of the inverse mapping from the two cameras are joined to a common ground [7] and generates a virtual top-down plan view of the world scene π.

By using the 4-point algorithm [9], the homography matrix H can correct the projective distortion of image planes I^1 and I^2 in Figure 3 and map on to the virtual top-down plan view that represents the world scene plane π. This virtual top-down view of the world scene is camera-independent and orthographic, and provides the objective measure of the actual scene dimensions without view-dependent appearance. The orthographic virtual top-down plan view also enables the generation of the navigation

trajectory profile of moving objects in terms of position, velocity, and acceleration on the world scene coordinate system.

3. Design and Implementation

The overall system diagram is depicted in Figure 4. The system starts from the camera calibration after deploying the cameras to specific locations. The calibrated and synchronized cameras (camer-1 and camera-2) keep capturing the synchronized image frames. Computer vision algorithms conduct perspective projection and planar homography mapping as explained in the previous section. Color-based object tracking algorithm [11,12,13] follows the processing, and the navigation-trajectory profiling is achieved by analyzing the results of the object tracking.

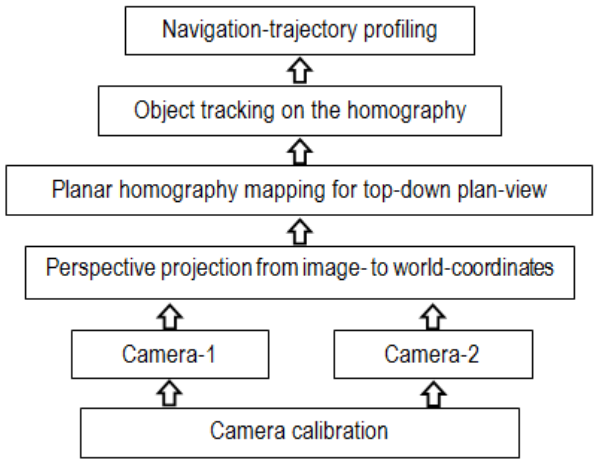

Figure 4. Overall System Diagram for the processes from imaging to trajectory profiling

The system diagram is implemented into a prototype testbed composed of two cameras, a frame-grabber board, a desktop computer, and a flat table as shown in Figure 5. The current testbed uses two synchronized cameras, and they can be mounted in versatile manner: for top-down view or arbitrary oblique views depending on user's need. In Figure 5, the left-view camera is installed on top of a tripod while the right-view camera was attached to a clamp fastened on a bookshelf cabinet. The flat table used as a prototype testbed.

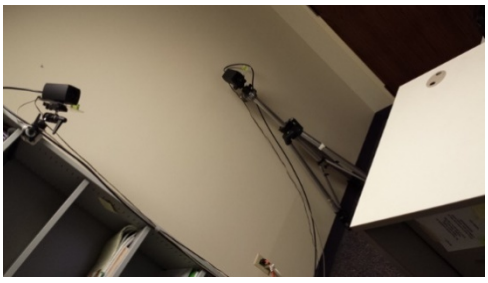

Figure 5. Testbed configuration using a flat table and two synchronized cameras. Left-view camera on the tripod, right-view camera on the clamp

The left- and right-view images captured by the cameras are shown in Figure 6. The appearance of the same flat table looks very different on the left- and right-view images due to the perspective effects of the two cameras. We define the world seen coordinate system on the flat table in terms of the origin O and X- and Y-axis extended from the origin as shown in the right side image of Figure 6. Note that it is conventional to define the image domain's Y-axis to extend toward row direction of the image (in Figure 6), while the mathematical definition of the Y-axis is opposite to it (in Figure 2.)

Figure 6. Left- and right-view images of the testbed. World coordinate system is defined on the testbed (overlaid for visualization on the right-view image.)

Figure 7. Snapshot of camera calibration using a standard calibration pattern

Camera calibration is achieved by using a standard checker board pattern shown in Figure 7. Multiple snapshots of the calibration board at different positions are used in the procedure.

Figure 8. Snapshot of pilot study using a small mobile robot (indicated by arrow) on the testbed

Figure 8 shows the left- and right-view snapshot of the experimental run of a micro-robot running on the flat table. The micro-robot is indicated by the arrows for readers' convenience. Individual trajectory points of the robots are time-stamped, and the calculation of the position, velocity, and acceleration provides the full description of the robot movements. We tested two different micro-robots of different running speed for the implementation of the prototype testbed.

The testbed has an intuitive user interface as in Figure 9 with which students can run the system easily and visualize the trajectories intuitively. The control software for the video capture and the display on GUI frontend in Figure 9 was written in C++ language. The prototype system was set up in the author's office space, but the system is scalable to deploy to larger environments such as a bigger lab floor for mini-robots or a spacious gym floor for bigger robots. Different deployment options do not require the software code modification; only the camera calibration needs update for the new camera positions.

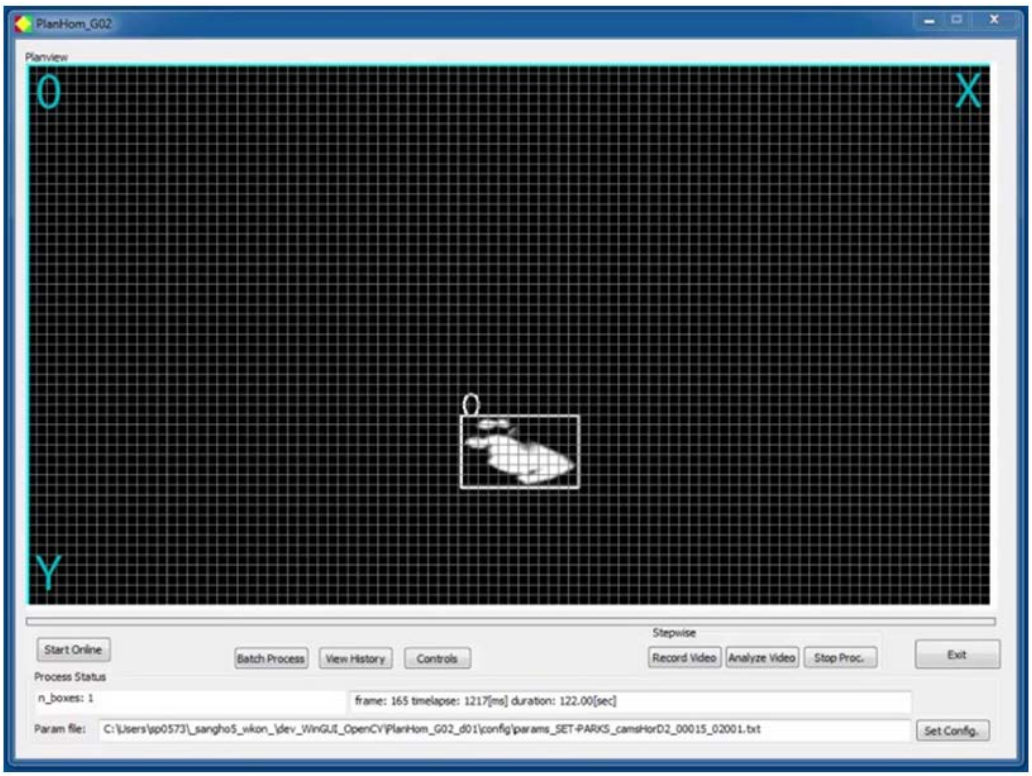

Figure 9. Graphical-user-interface frontend that controls the system and visualizes results. The overlaid world coordinate system of O, X, Y complies to that of the testbed in Figure 6

4. Experiments

The prototype testbed was extended at a larger scale and applied to real-world experiments. The extended testbed used in the experiment is composed of two cameras, a desktop computer, and a flat floor area as shown in Figure 10.

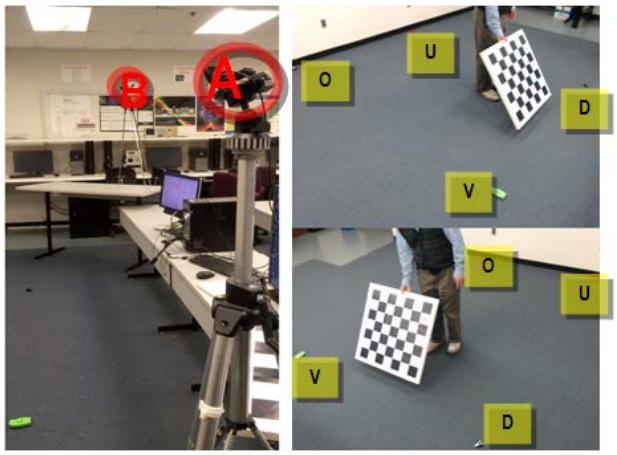

Figure 10. Testbed (left) using two synchronized cameras A, B and their views (right)

The two synchronized cameras are marked 'A' and 'B' for readers' convenience. The upper-right view in Figure 10 is from camera-A, while the lower-right view in Figure 10 is from camera-B. The labels O, U, V, and D are add-on for readers' convenience, and indicate the locations of the markers placed on the floor for the 4-point algorithm [12] to specify the user-defined world coordinate system.

The appearance of the same scene looks very different on the camera-A vs. camera-B images due to the different perspectives. We define the world coordinate system on the flat area in terms of the origin O and U- / V-axis extended from the origin. Camera calibration is achieved by using a standard checker board pattern shown in Figure 10. Multiple snapshots of the calibration board at different positions are used in the procedure.

Figure 11 shows a remote control car used in the experiment, and the two camera views of the car on the testbed. We used a remote control car with a known maximum speed (i.e., 8-12 miles per hour or 3.5 – 4.3 meters per second) in the experiment. The right-most picture of Figure 11 shows the plannar-homography mapping of the two camera views to achieve a virtual top-down view on the user-defined world coordinate system. The world coordinate system is defined by the horizontal axis between two points O and U (denoted by the horizontal line) and the vertical axis between O and V (denoted by the vertical line) in Figure 11.

The automatic tracking of remote-control car is achieved by color-histogram based object tracking algorithm [11,12,13]. The algorithm was chosen for its real-time processing speed and efficiency. This experiment assumes the remote-control car is distinct in color. The tracking algorithm generates a fitted ellipse overlaid on the tracked object (i.e., remote-control car) as shown in Figure 11 and Figure 12. The center of the ellipse comprises the individual trajectory point along the video frames. The individual trajectory points of the robot are time-stamped, and the calculation of the position, velocity, and acceleration provides the full description of the robot movements. It is straightforward to convert between image coordinate system (U,V) vs. mathematical coordinate system (X,Y) [9].

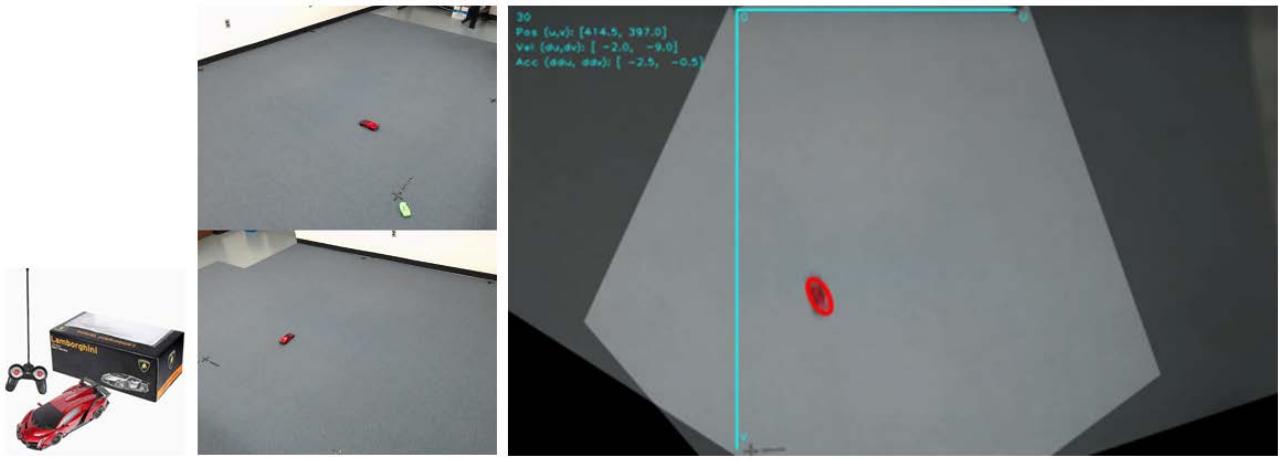

Figure 11. Snapshot of experiment using a small remote-control car in lieu of mobile robot on the testbed

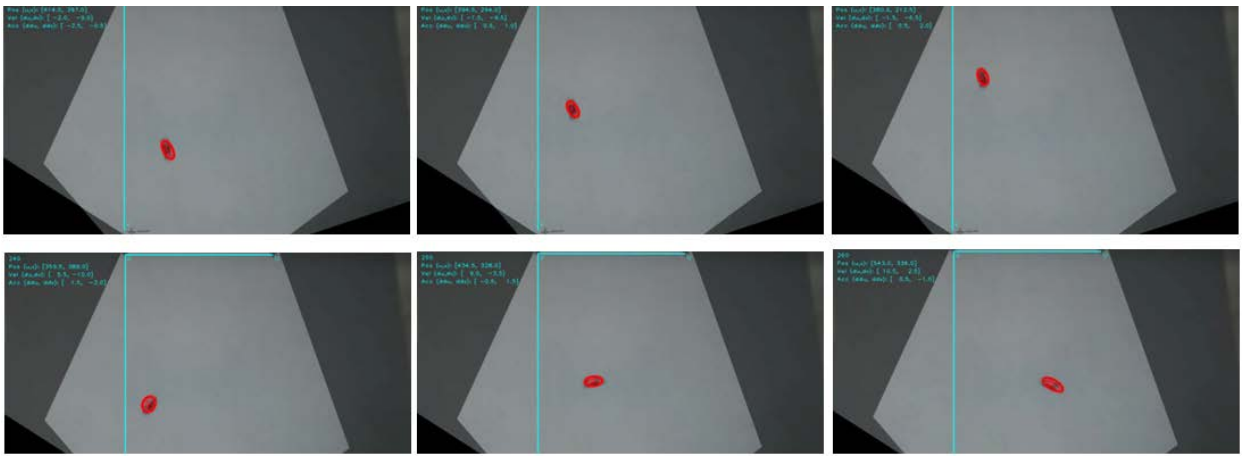

Figure 12. Examples of tracking: linear navigation (top row) vs. circular navigation (bottom row)

Figure 12 shows two instances of tracking: linear navigation (shown in the top row) and circular navigation (shown in the bottom row). Each homography-mapped frame contains the information of the frame number, the position, velocity, and acceleration of the tracked mobile car in the image coordinate units (i.e., pixels.)

The testbed has an intuitive user interface (Figure 5 and Figure 11) with which students can run the system easily and visualize the trajectories intuitively.

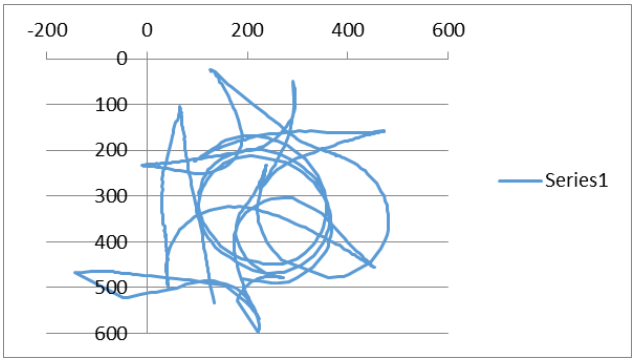

Figure 13. Example profile of a navigation trajectory

Figure 13 shows an example profile of a cumulative navigation trajectory that involves 1830 frames (including those in Figure 12.) The tracking algorithm of the control software successfully tracks the moving robot and assigns the enclosing ellipse with track ID as shown in Figure 11 and Figure 12. The center position (x_i, y_i) of the ellipse at the *i*-th frame is regarded as the position of the robot at the given time. The instantaneous velocity is defined as a vector $\vec{v}_i = (v_x, v_y)_i$ in terms of the differential location in x- and y-axis between the current *i*-th frame and the previous (*i*-1)-th frame. The velocity components v_x and v_y are defined in equations (7) and (8):

$$v_x = \frac{x_i - x_{i-1}}{\Delta t} \tag{7}$$

$$v_y = \frac{x_i - x_{i-1}}{\Delta t}. \tag{8}$$

The speed of velocity is defined by the amplitude of the vector in equation (9):

$$|v| = \sqrt{{v_x}^2 + {v_y}^2}. \tag{9}$$

Figure 14 shows the speed profile for the navigation trajectory of the Figure 13. The horizontal axis is frame number, while the vertical axis is speed in [mm/sec]. It shows multiple sharp spike noises, which are due to imaging noise and tracking error. But human interpreter can easily rule out those noise points from the speed profile. The typical value of the measured speed on the smooth bumps/plateaus of the Figure 14 profile is approximately 2 [meters/sec], which is reasonable in the current experimental setting.

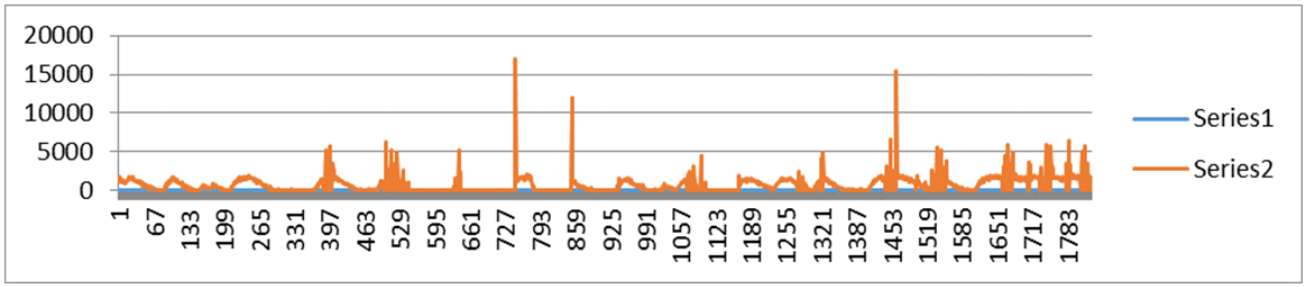

Figure 14. Speed profile for the navigation trajectory of Figure 13

The instantaneous acceleration is defined in equation (10) as the vector difference between the current *i*-th frame and the previous (*i*-1)-th frame:

$$\vec{a}_i = \frac{\vec{v}_i - \vec{v}_{i-1}}{\Delta t}. \quad (10)$$

The control software for the video capture, the object tracking, and the profile generation was written in C++ language. We tested the proposed system with a general-purpose desktop PC installed with Microsoft Windows 7 operating system and 3.4 GHz Intel Core i7 CPU. The experimental testbed captures two synchronized camera images at 30 frames per second, and the color-based tracking algorithm runs almost in real-time; tracking speed may vary slightly depending on the imaging quality. The current system was setup in a small lab space, but the system is scalable to deploy to larger environments such as a bigger floor for moderate-size robots or a spacious gym floor for bigger robots. Different deployment options do not require the software code modification; only the camera calibration needs update for the new camera positions.

The proposed system provides objective trajectory profiles of the robot navigation in terms of the position, velocity, and acceleration at every frame. Some inaccuracy is induced at some images; most of the disturbance in the accuracy is from imaging noise and tracking errors; it is an open research issue in computer vision to achieve a robust detection and perfect tracking.

5. Conclusion and Future Work

This paper presented a vision-based evaluation testbed for mobile robot navigation. The testbed can be used to estimate mobile robot navigation in quantitative manner and to provide students and developers with insights about objective performance evaluation based on the physics-based profile. The proposed system is a general-purpose system that can be deployed to estimate the navigation of various moving objects including, but not limited to, mobile robots. The developed system is versatile in that the cameras can be installed in various configurations such as top-down viewing direction or arbitrary oblique viewing directions as long as the two views are not in parallel. The system is non-obtrusive since it is purely vision-based and does not require the attachment of any sensors, transponders, or beacons to the tested object. The future research plan includes the improvement of imaging and tracking algorithms and testing the system in various environments such as tracking of multiple objects in a wide area.

Acknowledgements

This research was supported and funded by AAUP MR&R Research Grant from Central Connecticut State University.

References

[1] National Science Board. “Preparing the Next Generation of STEM Innovators: Identifying and Developing our Nation’s Human Capital,” 2010. Retrieved from: http://www.nsf.gov/nsb/publications/2010/nsb1033.pdf.

[2] Aggarwal, J.K. and Cai, Q. “Human motion analysis: a review”. Computer Vision and Image Understanding, 73(3): 295-304. 1999.

[3] Gavrila, D. “The visual analysis of human movement: a survey”. Computer Vision and Image Understanding, 73(1): 82-98. 1999.

[4] T.B. Moeslund, T. B. and E. Granum, E. “A survey of computer vision-based human motion capture”. Computer Vision and Image Understanding, 81(3): 231-268. 2001.

[5] Beinhofer, M. & Burgard, W. “Efficient Estimation of Expected Distributions for Mobile Robot Navigation”, Proc. of the Austrian Robotics Conference. 2014.

[6] Suliman, C., Cruceru, C. & Moldoveanu, F. “Mobile Robot Position Estination Using the Kalman Filter”, Scientific Bulletin of the Petru Maior University of Tirgu Mures, Vol. 6 (XXIII). 2009.

[7] Hartley, R. & Zisserman, A. Multiple View Geometry in Computer Vision, 2nd ed. Cambridge University Press. 2003.

[8] Forsyth, D. and Ponce, J. Computer Vision: A Modern Approach, Prentice Hall, 2003.

[9] Ma, Y., Soatto, S., Kosecka, J., & Sastry, S. An Invitation to 3-D Vision: From Images to Geometric Models, Springer Ltd. 2001.

[10] Criminisi, A., Reid, I. & Zisserman, A. “A plane measuring device”, Image and Vision Computing, 17(8):625-634, 1999.

[11] Allen, J.G., Xu, R.Y.D., & Jin, J.S. “Object Tracking Using CamShift Algorithm and Multiple Quantized Feature Spaces”, in Conferences in Research and Practice in Information Technology, Vol. 36. M. Piccardi, M., Hintz, T., He, X., Huang, M.L., Feng, D.D. & Jin, J. Editors. 2004.

[12] Bradski, G.R. “Computer Vision Face Tracking for Use in a Perceptual User Interface”, Intel, 1998

[13] Fukunaga, K. “Introduction to Statistical Pattern Recognition,” Academic Press, Boston, 1990.

[14] Park, S. and Trivedi, M. “Understanding Human Interactions with Track and Body Synergies (TBS) Captured from Multiple Views,” Computer Vision and Image Understanding: Special Issue on Intelligent Visual Surveillance, 111(1), pp. 2-20, Elsevier Inc. 2008.