A low-cost test-bed for real-time landmark tracking

Ambrus Csaszar, Jay C. Hanan, Pierre Moreels, Christopher Assad

Ambrus Csaszar, Jay C. Hanan, Pierre Moreels, Christopher Assad, "A low-cost test-bed for real-time landmark tracking," Proc. SPIE 6555, Sensors and Systems for Space Applications, 655511 (3 May 2007); doi: 10.1117/12.720183

Event: Defense and Security Symposium, 2007, Orlando, Florida, United States

A low-cost test-bed for real-time landmark tracking

Ambrus Csaszar,[a] Jay C. Hanan[*],[b] Pierre Moreels,[a] Christopher Assad[c]
[a]California Institute of Technology, 1200 E California Blvd., Pasandea, CA USA 91125
[b]Oklahoma State University, 700 N. Greenwood Ave., Tulsa, OK, USA 74078
[c]Jet Propulsion Laboratory, 4800 Oak Grove Dr., Pasadena, CA, USA 91109

ABSTRACT

A low-cost vehicle test-bed system was developed to iteratively test, refine and demonstrate navigation algorithms before attempting to transfer the algorithms to more advanced rover prototypes. The platform used here was a modified radio controlled (RC) car. A microcontroller board and onboard laptop computer allow for either autonomous or remote operation via a computer workstation. The sensors onboard the vehicle represent the types currently used on NASA-JPL rover prototypes. For dead-reckoning navigation, optical wheel encoders, a single axis gyroscope, and 2-axis accelerometer were used. An ultrasound ranger is available to calculate distance as a substitute for the stereo vision systems presently used on rovers. The prototype also carries a small laptop computer with a USB camera and wireless transmitter to send real time video to an off-board computer. A real-time user interface was implemented that combines an automatic image feature selector, tracking parameter controls, streaming video viewer, and user generated or autonomous driving commands. Using the test-bed, real-time landmark tracking was demonstrated by autonomously driving the vehicle through the JPL Mars yard. The algorithms tracked rocks as waypoints. This generated coordinates calculating relative motion and visually servoing to science targets. A limitation for the current system is serial computing—each additional landmark is tracked in order—but since each landmark is tracked independently, if transferred to appropriate parallel hardware, adding targets would not significantly diminish system speed.

Keywords: automation, autonomous driving, odometry, rover, target recognition, tracking, Mars yard, vision, waypoint.

1. INTRODUCTION

A low cost computer-controlled vehicle to serve as a platform for testing autonomous off-road navigation algorithms is presented. As a demonstration of the system and as a stepping-stone for further research, results of a high-speed visual landmark tracking algorithm used as a navigation aid on the vehicle are also discussed.

It is well-known that during the autonomous operation of a vehicle on sandy and rocky terrain, wheel slippage causes an ever-increasing degradation in the confidence of dead-reckoning localization (for example [1]). One way to boost localization confidence and keep the integral effects of odometry error in check is the tracking of static visual landmarks during the operation of the vehicle. Such robust real-time visual odometry has application for many kinds of automation.

In the context of slow, high-fidelity machines such as the Mars Exploration Rover (MER) pair designed and built at the Jet Propulsion Laboratory (JPL), a visual odometry system (called VisOdom) has been developed. It augments traditional odometry-based localization by the correlation of points in successive stereo image pairs taken at roughly half-meter intervals. VisOdom has exposed the utility of visual odometry in autonomous navigation in difficult terrain, allowing the rovers to make longer autonomous traverses. However, the MER rovers were not designed for visual odometry. Due to limitations in flight hardware and the numerical complexity of point correlation between images, VisOdom produces a considerable amount of down-time during which the rover cannot move. Furthermore, while it is being developed to aid obstacle avoidance [2] and instrument placement [3], limitations inherent to the current rover design and the nature of VisOdom force the rover to rely entirely on traditional odometry between pauses for image pair acquisitions.[4]

The present work has departed from traditional rover design in favor of creating a speedy, responsive, low-cost vehicle capable of utilizing and aiding in the development of real-time, bio-inspired algorithms. Finding that available commercial robots were generally designed for operation in smooth, planar environments and were not well-suited for our intended off-road terrain, an off-the-shelf remote control car chassis was utilized for the vehicle's base. It was outfitted with a USB video camera, inertial measurement unit, wheel encoders, servo control boards, and a laptop

[*] Jay.Hanan@okstate.edu; phone 918-594-8238; fax 918-594-8558; www.osu-tulsa.okstate.edu/hanan/

Sensors and Systems for Space Applications, edited by Richard T. Howard, Robert D. Richards
Proc. of SPIE Vol. 6555, 655511, (2007) · 0277-786X/07/$18 · doi: 10.1117/12.720183

computer with a wireless link. The total cost of the vehicle's hardware (excluding the laptop) is less than $1000, a price which makes it readily attainable by research labs seeking a low-cost research platform.

As a demonstration of the vehicle's capabilities, a real-time neural-network-based visual landmark tracking algorithm was adapted and tested. It was previously used for tracking targets such as facial features, and aircraft [5]. This particular algorithm was selected for its high operating speed and its robust ability to adapt to rotational and scale changes of targeted features. Furthermore, the algorithm is well-suited to parallelization on an FPGA, greatly reducing power consumption and nearly eliminating its CPU load profile. Such characteristics open up the possibility that the algorithm could benefit a future planetary rover mission to provide localization data between VisOdom updates. Of course, the algorithm's usefulness is not limited to augmenting VisOdom, and could be utilized as a stand-alone guidance or localization tool as demonstrated here.

2. METHODS

2.1. Vehicle hardware

The vehicle hardware has evolved through several implementations. Each is described in their respective reports [6]. The original version and current version are juxtaposed in Figure 1.

The original chassis of the vehicle was modified to carry additional onboard electronics and to support the extra weight of a laptop computer. Figure 2 shows a blow-up view of the system. An additional 6:1 ratio gearbox was added to increase torque and counteract the weight of the laptop computer, while still allowing for a top speed of approximately one meter per second. The original motor and 7.4v battery pack remain in use. The suspension system has been

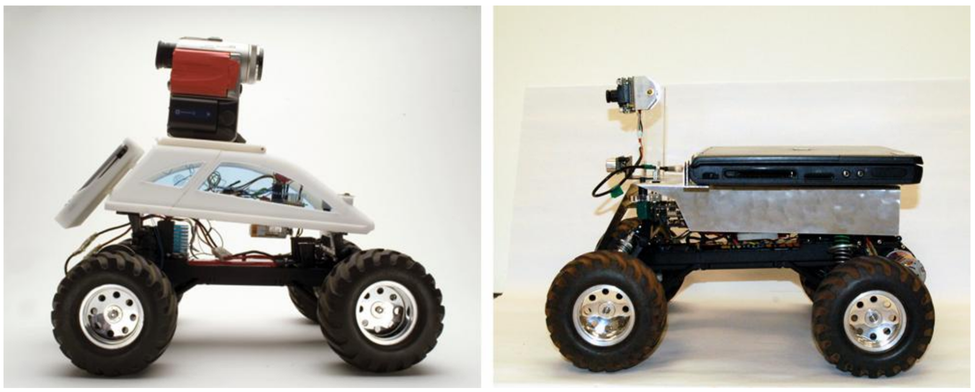

Figure 1 – *Left*: original palm-pilot driven vehicle with a hi-8 tape recorder. *Right*: current configuration with USB webcam and laptop computer.

replaced with stiffer adjustable springs and dampers in order to support the added weight of new components. Two Analog Devices ADXL103 [7] accelerometers oriented about the horizontal plane, as well as an Analog Devices ADXRS300 [8] single axis gyroscope oriented about the vertical axis has been installed. Two optical wheel encoders with 40 counts per revolution have been specially fabricated, one in each rear wheel. A commercially available Brainstem [9] servo control board replaces the original radio and is configured to relay motor and steering command values from the serial port (RS232) to the servos themselves. A second Brainstem board is dedicated to data collection from the wheel encoders and inertial measurement unit, and relays data at approximately 500 samples/sec to the serial RS232 port. The new electronics are powered via a 9 volt battery located in the chassis. An adjustable tilt USB camera is installed on a modified body shell. Finally, a laptop computer with a wireless link serves as the host for both the USB camera and the serial connection to the onboard Brainstems. The laptop need not be specific to the vehicle; it only needs to have the necessary USB and serial ports (a USB-to-serial adapter suffices) and a single Windows program (see below) installed in order to function.

2.2. Modular Software Architecture

The core of our design is the onboard module, which runs on the vehicle's laptop and handles all data acquisition and command relaying for the vehicle. Once initiated, this module sets up TCP servers that allow for the connection of separate control and data processing modules that can run either onboard the same laptop, or off-board on a separate computer system. For maximum speed while maintaining code reusability and readability, the onboard module is a Windows console application written in object-oriented C++.

A platform-independent GUI-based control module was written in Java, which can be run on an Apple, Unix, or Windows machine. It is designed for use on a separate work-station that is in wireless communication with the vehicle's onboard laptop, and allows the vehicle to function as a user-controlled data collection tool. Alternatively, the module can be used to observe (and, in the case of emergency, interrupt) the functioning of any automated control or data processing system written by the user.

Finally, the visual landmark tracking algorithm was modified via the use of a TCP message passing library, developed for interfacing with the vehicle's onboard module. The algorithm is currently under development at Oklahoma State University (OSU). It was first designed as an off-line tool to facilitate the analysis of recorded video, but during the course of this research it was modified to run on-line and process live video collected from the rover in near-real-time. Its front-end is a C++ Windows GUI program.

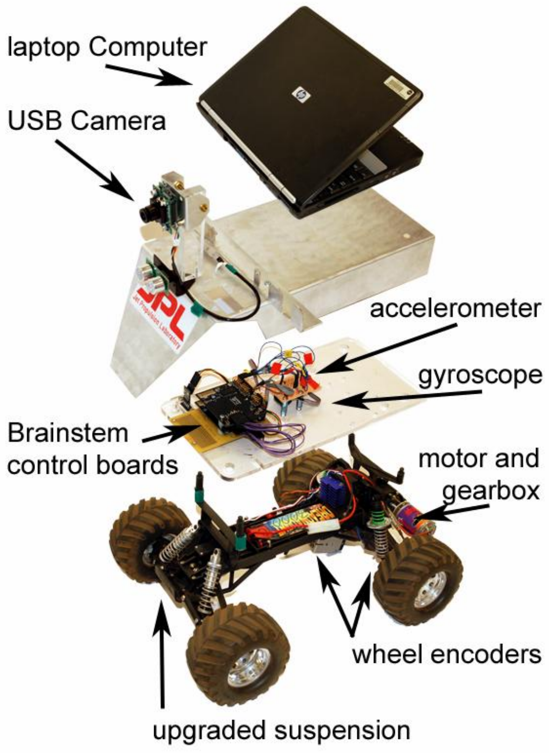

Figure 2 – A blow-up view of the vehicle hardware in its current configuration. Major modifications and added components are shown.

2.2.1. Onboard software module

The onboard module coordinates the retrieval of different types of data from several sources that operate at different frequencies. The vehicle must communicate with any data processing and control modules via TCP, read sensor information from and send commands to the Brainstem control boards via serial (RS232), and grab frames from the USB camera. All of these concurrent I/O operations require several threads of code to be executing in parallel while trading data back and forth.

The task is organized into three threads. Two threads deal with setting up TCP servers to which external modules may connect: one thread deals only with a data processing type connection, while the other thread handles control command type messages. The TCP data thread captures frames from the camera at 8-20Hz, and relays these along with sensor information from the rover to the connected data processing module. The TCP command thread receives servo and motor commands from a connected control module. The vehicle manager thread exchanges information with the Brainstems at the maximum speed that they can support (500Hz) and performs discrete-time integration of the IMU signal to produce an estimated velocity for the rover. In the duration of time between video frame captures, IMU data is integrated via a right Riemann sum of the form:

$$\underline{X}^{*}_{\mathrm{T_f}\text{-}\mathrm{T_i}} = \sum_{j=T_i}^{T_f} (\underline{X}_j)(t_j - t_{j-1}) \,, \tag{1}$$

$$\underline{X}^{*}_{\mathrm{T_f\text{-}T_i}} = \begin{bmatrix} \dot{x}_{T_f - T_i} \\ \dot{y}_{T_f - T_i} \\ \theta_{T_f - T_i} \end{bmatrix}, \quad \underline{X}_t = \begin{bmatrix} \ddot{x}_t \\ \ddot{y}_t \\ \dot{\theta}_t \end{bmatrix},$$

where X^* contains the average velocity and final relative heading for the time interval ($T_f - T_i$) between the capture of the previous video frame and the most current video frame. Each time a new video frame is captured, the value of X^* is retrieved by the TCP data thread, sent to any connected external modules, and re-zeroed.

2.2.2. Control software module

An off-board module written in Java provides a platform-independent high level control interface for the vehicle as well as the observation and logging of video and other sensor data in real time. Both manual and autonomous vision-guided control systems were implemented. The module also provides a front-end for several visual feature detectors that an operator may use to aid the selection of suitable visual landmarks.

2.3. Initial feature selection

In order to avoid tracking all points in the image, a small number of privileged points or features were selected. The underlying assumption is that objects in the image, consist of a collection of parts [10, 11, 12]. In observed images of the objects, these parts generate features. Figure 3 displays the features identified by the commonly used difference-of-Gaussians detector [13, 14, 15] on several images of the same rock face. Some features and groups of features are geometrically consistent with each other across different views of the face, since they are detected repeatedly near the same physical location. The crevices generate repeatedly multiple features; the corners of the rock, the upper edge, the shadow also generate features in each image; the broad regions, which are very smooth with little texture, do not generate many features in any view. Furthermore, since the background is different from one image to the next, the locations of the background feature detections are also completely different between any two images. Object recognition methods based only on such features-based geometric considerations have been developed successfully, for example by Lowe [16] or more recently by Fleuret and Geman [17]. In both cases the authors use perceptual grouping of simple features to form objects. In the present study we used three different interest point detectors: the Harris detector [18], which frequently selects interesting points near corners of objects; the Hessian detector [19], which also favors corners, but is based on second-order image derivatives versus first order derivatives for the Harris detector; and the difference-of-Gaussians, which frequently detects blobs and circular patterns. Comparisons of feature detectors have been performed in [20], and the performance of these three operators was similar.

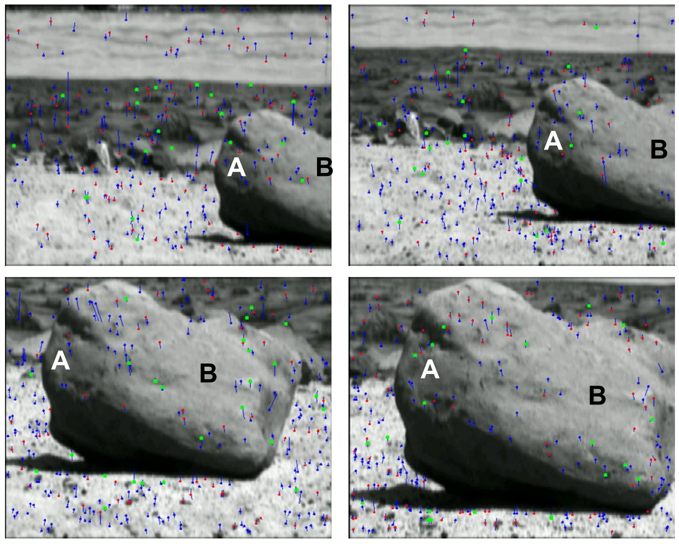

Figure 3 – Foreground features vs. background (from camera shown in Figure 1-left). A significant fraction of the features generated by the rock face are found again at the same location in separate views (A). Since the background changes, background features are generated at different locations in each picture. Note the consistency in lack of features about region "B."

While feature detectors are concerned with finding locations of interest, complementary information is provided by image descriptors which depict the local image texture or appearance near these points of interest. One of the simplest descriptors consists of selecting a patch of pixels around the detected point of interest (for example, a 10 × 10 patch), sampling all pixels in this patch, and aggregating them in a long vector. This representation is easy to compute and has been used widely in template matching, but it is sensitive to noise and to changes in lighting and viewpoint conditions, such as rotation or scale changes. To date, the most successful feature descriptors [20] are based on histograms of gradients like the popular SIFT [15]. In this study we used a representation similar to SIFT, but where the gradients are not sampled on a rectangular grid but with a log-polar grid. This provides improved invariance to rotation and emphasizes the importance of the center of the feature with respect the rest of the patch of interest. In general, pairs of features with a low distance in appearance space denote the same physical object part, although for large databases of features unfortunately many mismatches occur due to insufficient distinctiveness of the descriptor.

2.4. Visual servoing controller

During live tracking of one or more features, the operator may enable a proportional steering controller which can be used for a vision-guided approach to a waypoint. It attempts to aim the vehicle towards the centroid of the tracked feature set using a first-order approximation for the vehicle's angular deviation from desired course. The steering output angle, α, is given by the proportional relation:

$$\alpha_t = k_1 k_2 \sum_{i=0}^{N} \frac{x_{i,t-1}}{N}, \tag{2}$$

where $x_{i,t-1}$ is the horizontal displacement of the *i*-th tracked feature from the center of the most current video frame; k_1 is the proportional gain of the controller; k_2 is the angle subtended by a single pixel in the video camera's field of view. The "true" value of k_2 deviates from our constant near the edges of the video frame, due to higher-order camera lens distortion that can be approximated. [21]

2.5. Visual distance estimation

One method of generating localization data from visual information is the tracking of two distinct features whose initial location in space relative to the vehicle is known. The MER rovers can acquire such information from a 3-dimensional terrain map generated by image correlation from their stereo mast cameras.[22] Although the vehicle described in this research has no such capability, we operate under the premise that high-speed visual distance estimation could be used on hardware with similar still-image stereo cameras to those on MER. Thus, we present a method for monocular distance estimation given that the initial location of at least two suitable features is known relative to the vehicle, as shown in Figure 4.

For our calculations, we use the same first-order approximation for the angle subtended by a single pixel as in (3). Given the apparent pixel distance, (x_2–x_1) between the two tracked features, we can find the initial angle, β, which separates them:

$$\beta = k_2 (x_2 - x_1) . \tag{3}$$

Let us now refer to the centroid of the two features as the "waypoint" for the vehicle. Knowing β and the initial distance, *D*, along a straight path to the waypoint, we can approximate linear distance, *A*, that separates them:

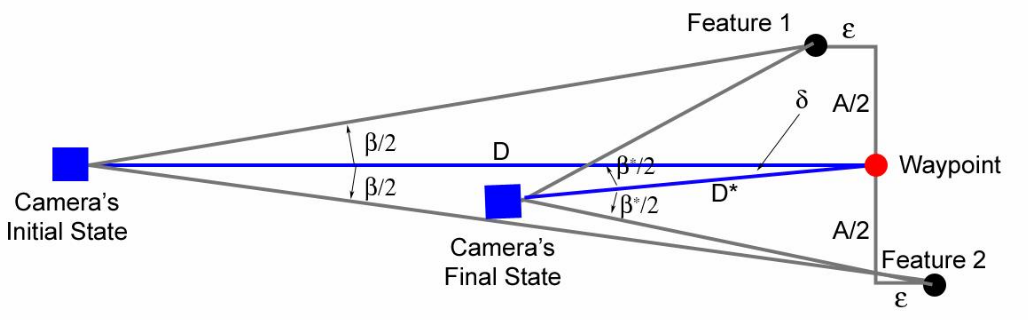

Figure 4 – Graphical representation of the geometry involved in estimating D*, Equation (5), based on Feature 1 and Feature 2.

$$A = D[\tan(\beta)]. \tag{4}$$

Finally, we can approximate any distance D^* along a straight-line path to the waypoint:

$$D^* = \frac{A}{\tan(\beta^*)}. \tag{5}$$

For small β, $\tan(\beta) \sim \beta$ and D^* becomes:

$$\frac{C}{x_2 - x_1} \tag{6}$$

Where $C = A/k_2$ and scales with target size.

D^* will provide an adequate approximation for the linear distance remaining to the waypoint, with the caveat that $\delta << \beta$ and $\varepsilon << A$ both must hold. Given that our vehicle platform has a forward-facing camera with a field of view less than 30 degrees, and is actively steering towards the waypoint during operation, it is likely that one feature will be lost from the field of view and tracking will fail before this approximation breaks down.

3. RESULTS AND DISCUSSION

We conducted testing in the JPL Mars Yard, an outdoor terrain designed to emulate landscapes found during the Viking 1 and 2 missions, as well as those encountered by MER. We present results from multiple traverses along two straight-line paths in the Mars yard.

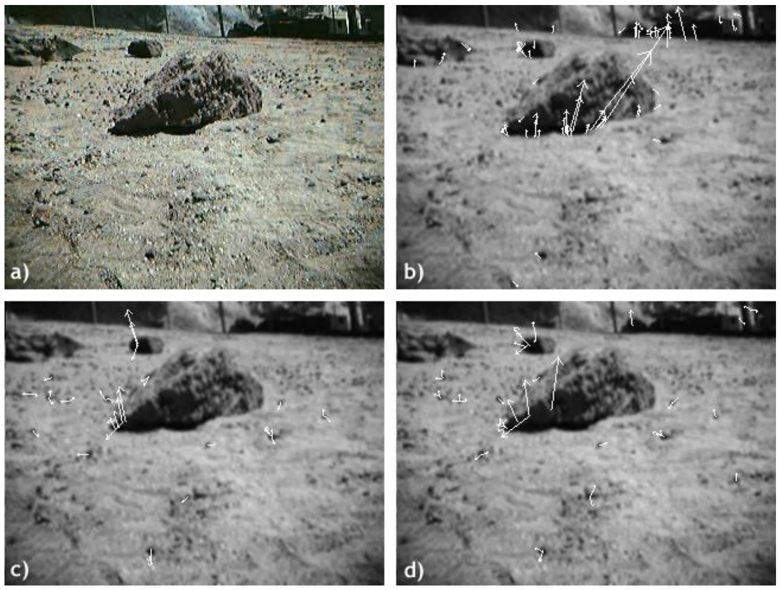

Figure 5 – An image from the vehicle's (Figure 1-right) camera analyzed with various feature detectors. Each vector indicates the location, orientation, and relative size of an extracted feature; a) the original image b) multi-scale Harris detector c) multi-scale Hessian detector d) multi-scale Laplacian detector.

3.1. Feature selection

A utility for automatically selecting initial features of interest was used as a visual aid to facilitate manual feature selection. Figure 5 shows the results of several types of detectors on a video frame captured by the vehicle's camera in the Mars yard before a live tracking run. This method of feature detection shows promise in its ability to quickly highlight many information-rich areas of an image, and could be used in future research for automatic feature acquisition for live tracking.

3.2. Tracking consistency

Since the tracking algorithm may autonomously learn new targets, it is possible to set conditions that confuse the rover such that it sees two targets that are similar and cannot distinguish them for a continued traverse. Setting a threshold for the confidence given by the neural network provides some safeguard against erroneously driving to the wrong target. However, determining this threshold is a subject of further study.[5] Since the rover is programmed to steer toward the target and the frame rate can be maintained such that the target remains in the region of search, a confidence threshold was not used for the test runs. For some runs this caused difficulty when the rover was manually accelerated at a rate significant enough to move the target out of the region of search between frames. In the case of the sequence shown in Figure 6, the frame rate was reduced to 5 fps and the acceleration was pressed to full throttle. The consequence of depending on feature descriptors with insufficient distinctiveness for the remaining available targets is confusion. Thus, a nearby rock was identified as similar to the original target rock. A solution to this beyond a network confidence threshold is improving the throttle control to eliminate Z-axis rotation of the rover chassis. In addition, horizontal stabilization of the camera (active or passive tilt), improved camera resolution and field-of-view would also extend the range over which the rover can rotate before losing a target.

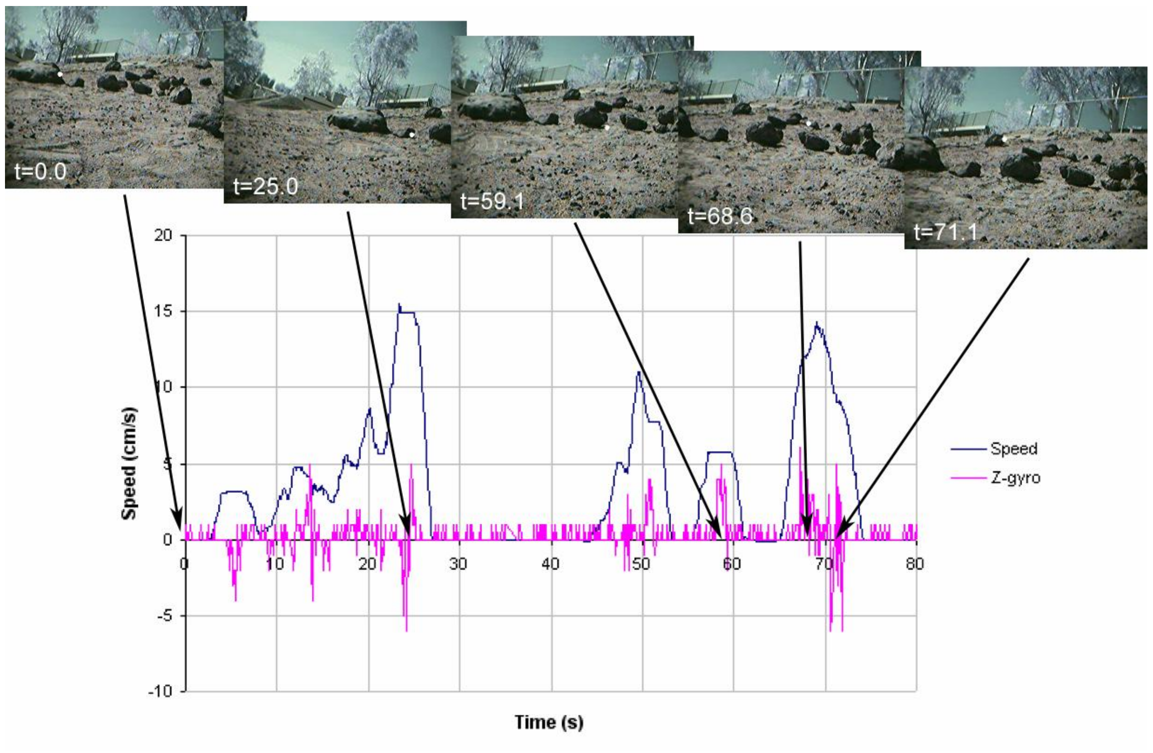

Figure 6 – The chart above shows the vehicle drive speed and un-integrated z-axis gyroscope data vs. time during a slope traverse. A sequence of frames from the traverse shows the tracking of a single feature indicated by a white dot in each image (5 fps). The tracking target was manually selected in the initial frame. Subsequent frames depict the state of the target tracker immediately after a tracking discontinuity has occurred. These tracking discontinuities are correlated with Z-axis rotational disturbances experienced by the vehicle chassis.

3.3. Visual servoing

Driving toward various targets in the Mars Yard was successful. Slipping of the wheels was later detected through the wheel encoders and sometimes observed directly. Such slipping would lead to serious errors in position estimation without the visual servoing. For example, Figure 7 is an example of a case where driving toward a waypoint was successful. Frames from a successful run with visual servoing, annotated with "perceived target" of the rover are shown.

For this run the frame rate was maintained above 8 fps and the travel velocity was on the order of 1 m/s (as indicated by encoders).

In-the-loop control of the rover steering has the added benefit of improving the data collected by the rover for post processing. Compared to manual control, the driving software is better able to keep the target rock in the field of view. An example of the steering corrections as seen from the recorded view is shown in Figure 8. These results are typical of data collected in the Mars yard.

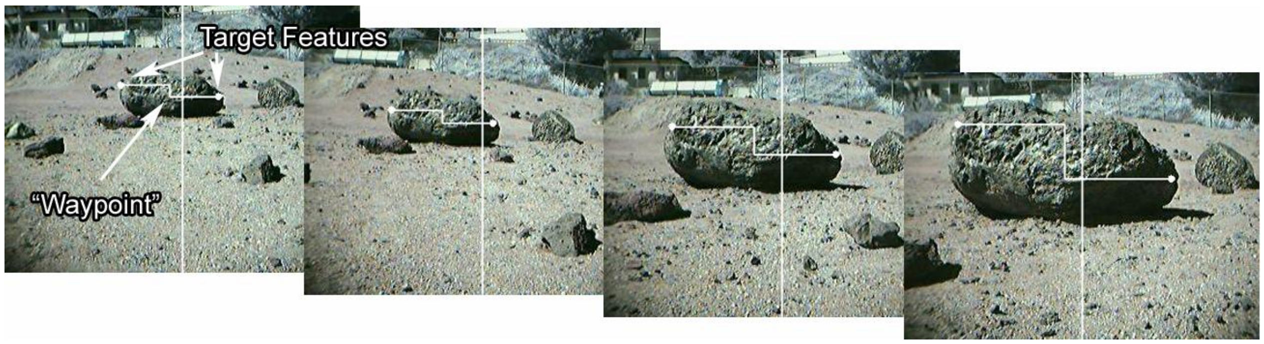

Figure 7 – A series of frames is selected from a successful approach to a waypoint. The centerline of each frame has been marked in order to show its changing relation to the waypoint.

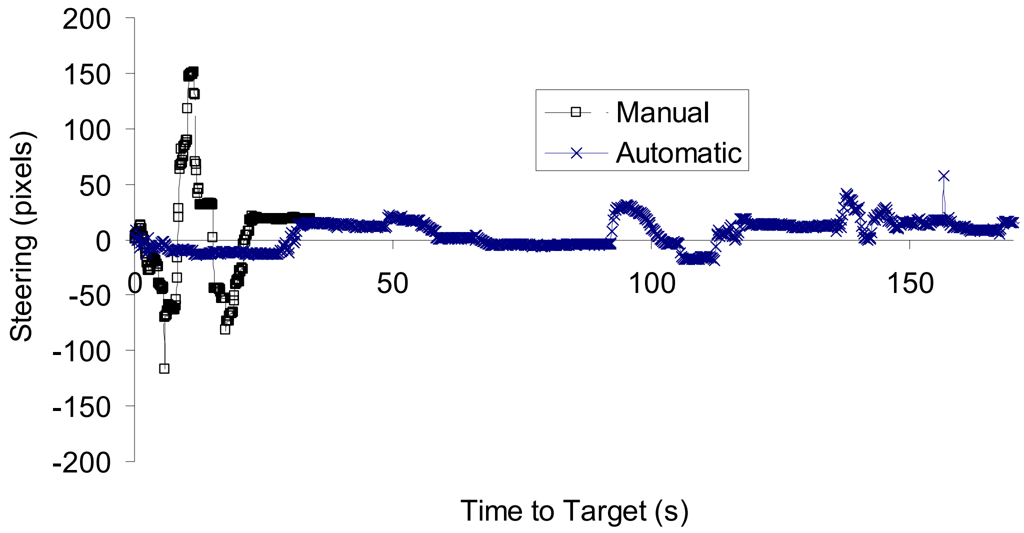

Figure 8 – Correction of steering toward the target based on visual odometry. A typical run from automatic steering is compared to a typical pattern in similar terrain calculated from post processing on frames recorded during manual steering. Automatic steering required lower magnitude corrections. Manual runs were generally shorter than the automatic runs as they were primarily used for testing system configurations. The target position is at 0 seconds.

In practice, the first order approximation of k_2 in equation (3) was adequate to determine the safe stopping distance, D_f, for a science target or waypoint. In our examples D_f could be determined to within 0.002 m in the lab using only single camera visual odometry. The target arrival position in the Mars Yard was accurate to within the resolution of manually performed position measurements (<0.02 m).

For this loose and rough terrain visual servoing was necessary to autonomously remain on course and determine arrival at a final destination. In most cases, surrounding hazards would have been encountered and target destinations would have been visually lost without the corrections provided from tracking waypoints.

4. CONCLUSIONS

The low cost system was successfully tested and tracked targets in the field for live visual odometry. The problem of automatic driving was addressed with a specific solution for automatically steering to arrive at selected targets.

Achieving success with the system required interaction of several components. Java provided a useful tool to implement the user interface and command the rover to begin an autonomous traverse.

Slipping was a significant error (often >50%) in dead-reckoning yet was detected and corrected using visual odometry. Based on tests for short traverses, an extension to longer more difficult traverses is promising. Key factors for a successful traverse with the present system include the selection of waypoints and moderation in acceleration toward a target.

4.1. Future Work

Targets were generally lost only when the rover accelerated forward abruptly. This tended to jostle the chassis and consequently the camera at high angular rates. Faster video processing and a better camera could alleviate this—leading to faster successful traverse rates. A new rover is in development at Oklahoma State University. This next version includes improved speed control, higher frame rate and camera resolution.

Feedback to the user on network generalization and confidence would also be helpful. Such data could be fed into the automatic target selector to determine frequency of selection. Automatic target and feature selection is envisioned for replacement of features that have gone out of the camera's field of view.

Higher fidelity hardware for the purpose of comparison with dead-reckoning is planned. High-resolution truth data will also improve performance analysis. A review and improvement of the software is underway. Lastly, a goal of the authors is implementation of the software in hardware for demonstration of low power consumption and minimization of processing demands.

5. ACKNOWLEDGEMENTS

Pavan Kayathi at Oklahoma State University provided support for waypoint tracking analysis (see also [5]). Casey L. Hughlett of Zion Labs, Inc. was instrumental in providing software development support. The SURF program at Caltech and the NASA Space Grant program provided student opportunities for building the low-cost rover hardware and collecting data for the experiments. The following students participated: Salomon Trujillo, Matthew King, and Ambrus Csaszar. The authors would also like to thank Pietro Perona at the California Institute of Technology and Tien-Hsin Chao of the Jet Propulsion Laboratory for their support. A portion of this research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration and funded through the Director's Research and Development Fund program.

6. REFERENCES

1. C. F. Olson, L. H. Matthies, M. Schoppers, M. W. Maimone "Rover navigation using stereo ego-motion." *Robotics and Autonomous Systems* **43** (2003) 215–229.
2. I.A. Nesnas, M. Bajracharya, R. Madison, E. Bandari, C.G. Kunz, M. Deans, M.G. Bualat, "Visual Target Tracking for Rover-based Planetary Exploration," *IEEE Aerospace Conference*, Big Sky, Montana, March 6-14, (2004).
3. M. Bajracharya, A. Diaz-Calderon, M. Robinson, M. Powell, "Target Tracking, Approach, and Camera Handoff for Automated Instrument Placement," *IEEE Aerospace Conference*, Big Sky, Montana, March (2005).
4. Y. Cheng, M. Maimone, L. Matthies, "Visual Odometry on the Mars Exploration Rovers," *IEEE Robotics and Automation Magazine* Special Issue (MER), **13**(2), June (2006), 54 - 62.
5. J. C. Hanan, P. Kayathi, C. L. Hughlett, "Position Estimation and Driving of an Autonomous Vehicle by Monocular Vision." *SPIE Optical Pattern Recognition* **18** (2007).

6. C. Assad, T. H. Chao, J. C. Hanan, A. Csaszar, P. Moreels, P. Perona "Real-Time Landmark Tracking." *JPL DRDF Annual Report* (2006), http://drdf.jpl.nasa.gov/reports/reports-public.cfm
7. Analog Devices Company Website, "ADXL103 Precision ±1.7g Single Axis iMEMS® Accelerometer", retrieved March (2007) from: http://www.analog.com/en/prod/0%2C2877%2CADXL103%2C00.html
8. Analog Devices Company Website, "ADXRS300 ±300°/s Single Chip Yaw Rate Gyro with Signal Conditioning", retrieved March (2007) from: http://www.analog.com/en/prod/0%2C2877%2CADXRS300%2C00.html
9. Acroname Robotics Company Website, "Brainstem GP 1.0 Module", retrieved March (2007) from: http://www.acroname.com/robotics/parts/S1-GP-BRD.html
10. M.A. Fischler, R.A. Elschlager, "The representation and matching of pictorial structures." *IEEE Transactions on Computer*, c-22, 1 Jan. (1973) 6792.
11. R. Fergus, P. Perona, A. Zisserman, "Object Class Recognition by Unsupervised Scale-invariant Learning", *IEEE. Conf. on Comp. Vis. and Patt. Recog.*, (2003).
12. P. Moreels and P. Perona, "Common-Frame Model for Object Recognition." *Proc. Neural Information Processing Systems*, (2004).
13. J.L. Crowley and A.C. Parker, "A representation for shape based on peaks and ridges in the deference of low-pass transform", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6** (1984) 156-168.
14. T. Lindeberg, "Scale-space theory: a basic tool for analising structures at deferent scales", *Journal of Applied Statistics*, **21**(2), (1994) 225-270.
15. D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints." *International Journal of Computer Vision*, **60**(2), (2004) 91-110.
16. D.G. Lowe, "Perceptual organization and visual recognition." *Kluwer Academic*, (1985).
17. F. Fleuret and D. Geman, "Coarse-to-fine face detection." *International Journal of Computer Vision*, **41** (2001) 85-107.
18. C. Harris and M. Stephens, "A combined corner and edge detector." *Alvey Vision Conference*, (1988) 147-151.
19. P.R. Beaudet, "Rotationally invariant image operators," *International Joint Conference on Pattern Recognition*, Kyoto, Japan, (1978) 579-583.
20. P. Moreels and P.Perona, "Evaluation of Features Detectors and Features Descriptors based on 3D objects." *International Journal of Computer Vision*, (2006).
21. Bouguet, Jean-Yves, "Camera Calibration Toolbox for Matlab", retrieved March 2007 from: http://www.vision.caltech.edu/bouguetj/calib_doc/
22. J.N. Maki, *et al.*, "Mars exploration rover engineering cameras," *Journal of Geophysical Research*, **108**, E12, (2003) 12-1–12-24.