Closing the Recognition Loop: Recognizing and Searching for Objects in the Real World

Jason Meltzer
Computer Science Department
University of California, Los Angeles
Email: jasonm1@cs.ucla.edu

Alberto Pretto
Department of Information Engineering
University of Padova, Italy
Email: alberto.pretto@dei.unipd.it

Brian Taylor and Stefano Soatto
Computer Science Department
University of California, Los Angeles
Email: {btay, soatto}@cs.ucla.edu

I. Introduction

In 1966, psychologist J. J. Gibson wrote, "we see because we move; we move because we see." While the latter statement is often taken for granted, the former is important, as well. We have developed a robotic system that can find and identify objects *whether or not they are initially visible* and update knowledge of the locations of such objects. To that end, our system combines a number of key components: *visual-inertial Structure from Motion with topological map-building* to localize the robot and map the environment, *real-time occlusion detection* to guide an efficient search of the environment, *object recognition/categorization*, and *path planning/navigation* to guide low-level control.

This abstract will briefly describe the overall system and these components, with references to longer works for details. A real-time demo will be on-site at the workshop.

II. System Components

We have incorporated a number of new and existing components in the development of this system. They are briefly described here.

A. Vision + Inertial SFM and Topological Map-Building

In order for the robot to navigate, it must be able to determine its pose in space and the relative positions of obstacles in its vicinity. For this task, we build upon an existing system developed by [3] that fuses visual information and inertial measurements in a monolithic structure from motion (SFM) + inertial measurement unit (IMU) extended Kalman filter. The *Corvis* system determines at any time the robot's position and orientation, as well as the 3D positions of a sparse set of points, relative to its initial reference frame. This information allows the robot to localize itself and detect obstacles in its environment. A topological map is built along with the geometric SFM, which allows the system to locally preserve fine-scale geometry while globally maintaining a coarse map of distant locations. Loop detection and closure is achieved by storing SIFT descriptors of the environment, then using a bag-of-features approach to recognize locations when the egomotion indicates a loop closure is possible.

Unified SFM+IMU: [3] demonstrates an observable extended Kalman filter for fusing visual and inertial measurements with bounded bias and automatic camera-IMU calibration. The system augments [2], which presents a real-time causal structure from motion system based on a similar filter. Both incorporate visual data in the form of *feature tracks* – sparse sets of points on the image plane that are continually measured – and the former adds linear acceleration and angular velocity measurements from an IMU.

Of particular importance for the current work is the continual updating of egomotion $(R(t), T(t))$ and the determination of the 3D structure of the environment (X) relative to the initial reference frame. This information is directly used by the robot's navigation system to localize itself and avoid obstacles when path planning.

B. Occlusion Detection

To find objects not immediately visible, the robot must know where to look. Rather than randomly searching for objects, the robot explores areas of the environment that likely to hide the target, namely behind *occlusions.* To do so, we adopt the strategy of detecting *occluding boundaries* and navigating around them.

Occluding Boundaries: An *occlusion* is a surface in the scene that lies between the camera in its current pose and a more distant surface. We determine which portions of the scene may be occluded by looking for *occluding boundaries,* which are 1D visual structures arising where the visible portions of two surfaces at different depths project to the same locations in an image. It is not possible to detect occluding boundaries with a single image, but motion of a single camera (or the use of a stereo camera) can reveal these boundaries. We have developed a novel real-time occlusion detection algorithm that finds the image coordinates of occluding boundaries with a single moving camera. The locations of these boundaries are provided to the navigation system to plan paths around them.

Occluding Boundary Detection: Our approach leverages feature tracking [1] and techniques from descriptor generation and matching [5] to find portions of the scene that contain possible occluding boundaries. We begin with the following observation: when viewing a continuous Lambertian surface, relative motion of the camera induces a continuous diffeomorphic warping of the image. [8] This is the fundamental principle behind most feature descriptors used for visual matching. However, at an occluding boundary, where the surface is not continuous, no such warping is possible. One can thus determine the position of an occluding boundary by

finding where this principle is violated.

To begin feature tracking, we select points using the method of [6] on the first image and when more features are required. An affine tracker [1] tracks these points over time, returning their new image-plane positions, as well as orientations, scales, and skews relative to the initial frame. We place a circular occlusion detection region of radius σ around each feature. This region is broken into a spatial grid in the manner of SIFT ([5]), and we construct a multi-dimensional histogram of both normalized intensities and weighted gradient orientations on this grid. On subsequent frames, feature points are tracked and these regions are warped according to an affine warping determined by the tracker. A new histogram is built, which can be compared with the statistics of prior histograms. Within a spatial bin that contains an occluding boundary, a large discrepancy will be observed over time within that bin's histogram. A statistical test is used to determine which spatial bins contain occluding boundaries. These are then searched for strong edges, which are flagged as occluding boundaries. This information is continually updated and supplied to the navigation system for path planning.

C. Object Recognition

We use a modified type of *bag of features* to detect and recognize objects, similar to [4]. Since our system seeks to recognize and localize objects, we do not want binary classification of an entire image, but rather recognition confidence in image sub-windows. This allows the robot to localize the object of interest in its map and to boost its confidence of recognition by moving toward and around the object until confidence is sufficiently high.

We combine the scale-invariant selection technique of [5] and windowed bags-of-features to classify image regions as particular objects with some level of confidence (the TF-IDF score). The system only needs to do scale and orientation selection once per feature, since we update a region's scale and orientation based on the egomotion reported by the SFM-IMU system. This contributes greatly to efficiency as scale selection is computationally intensive.

D. Path Planning and Navigation

The output of SFM-IMU, occlusion detection, and recognition all converge in the path planning and navigation module. Standard techniques are used to compute a path through configuration space and to command the drive motors. The set of 3D feature points estimated by the SFM module are projected in a 2D occupancy grid map that holds the probabilities that cells are occupied by obstacles. Only features that fall within a minimum and maximum height are projected into the grid. The probabilities of the cells that lie inside the current filed-of-view are cyclically decreased, thus features observed only briefly will vanish from the grid.

A binarized version of the occupancy grid is supplied to the NF1 planner (or *grassfire*): at every time step, we perform a wavefront expansion inside the free space of the grid from the goal cell to the cell representing the robot's position. Every free cell is marked with its distance to the goal cell, and the planner computes a solution trajectory by linking adjacent cells closest to the goal. In order to drive the robot to the goal position, we define a set of intermediate positions lying on the path. The robot then follows the planned trajectory using a stabilizing feedback control for differential-drive robots. [7]

A novel exploration strategy is used to search for objects. First, the robot scans the environment that is immediately visible, using the object recognition module to determine if any recognizable object may be in view. Any such candidate is immediately explored further by moving in its direction until a determination of its identity can be made with high confidence. Whenever the robot moves, the occlusion detection module indicates locations of occluding boundaries, which are stored in the navigation map. When no candidate objects are in view, the robot explores the space *behind occluded portions of the environment* by navigating around occluding boundaries. The navigation map stores what portions of the environment have been explored by the robot's camera and the locations and identities of objects recognized. The robot finishes exploration when the entire workspace has been explored.

As the map is stored on disk, the robot may revisit any object at some later time. Future work will explore ways to update this map to account for moved or manipulated objects.

E. Hardware

The robot's hardware consists of the following devices:

Camera: A wide-angle FireWire camera (1024x768, 30fps) is used for all vision on the robot.

BEI C-MIGITS III: The IMU provides high-resolution linear acceleration and rotational velocity at 100Hz.

Evolution Robotics ER1 Control Module: The ER1 control module and stepper motor wheels provide mobility through an interface with the Player/Stage software.

Computing: A MacBook Pro/4GB is used for all computing.

Point Grey LadyBug II: While not currently used, a Point Grey LadyBug II 6-camera color omnicam is installed on the robot. It will be used for object recognition and occlusion detection in future versions of the system.

REFERENCES

[1] S. Baker, R. Gross, and M. I. Lucas-kanade 20 years on: A unifying framework: Part 4. Technical Report CMU-RI-TR-04-14, CMU, Pittsburgh, PA, February 2004.

[2] H. Jin, P. Favaro, and S. Soatto. Real-time 3-d motion and structure from point features: a front-end system for vision-based control and interaction. In *Computer Vision and Pattern Recognition; code available from* `http://vision.ucla.edu`, pages 778–779, June 2000.

[3] E. Jones and S. Soatto. Visual-inertial navigation, localization and mapping: A scalable real-time large-scale approach. *Technical Report UCLA-CSD-0900xx*, August 13, 2009.

[4] C. Lampert, M. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1 –8, 23-28 2008.

[5] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2(60):91–110, 2004.

[6] J. Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[7] R. Siegwart and I. R. Nourbakhsh. *Introduction to Autonomous Mobile Robots*. The MIT Press, 2004.

[8] A. Vedaldi and S. Soatto. Features for recognition: viewpoint invariance for non-planar scenes. Technical Report, October 2005.