Improving Visual SLAM Algorithms for use in Realtime Robotic Applications

Patrick Benavidez, Mohan Kumar Muppidi, and Mo Jamshidi, Ph.D.Lutcher Brown Endowed Chair Professor
Autonomous Control Engineering Lab
Department of Electrical and Computer Engineering
University of Texas at San Antonio
San Antonio
USA
patrick.benavidez@gmail.com, mkumar2301@gmail.com, moj@wacong.org

***Abstract*—Many vision-based Simultaneous Localization And Mapping (vSLAM) algorithms require large amounts of computational power and storage. With these requirements, vSLAM is difficult to implement in real time. One known bottleneck in vSLAM is performing feature identification and matching across a large database. In this paper, we present a system and algorithms to reduce computational time and storage requirements for feature identification and matching components of vSLAM. We compare our algorithms using ORB and SURF to their unmodified versions readily available datasets and show significant reductions in storage requirements and calculation time.**

Keywords—indoor robot, vSLAM, cooperative SLAM

I. INTRODUCTION

In an ideal robot navigation scheme, sensory information is processed immediately to create a control vector for the actuators to guide the robot to its target while completely avoiding danger. Important to an ideal navigation scheme are two concerns: the speed of processing sensory data and processing of a high level of environmental information. With vision based Simultaneous Localization And Mapping (vSLAM), a high level of environmental information in the terms of landmarks is obtained. What is not guaranteed by SLAM is the speed of processing.

Algorithms such as Scale Invariant Feature Tracker (SIFT) [1], Speeded-Up Robust Features (SURF) [2], Features from Accelerated Segment Test (FAST) [3], and Oriented FAST and Rotated BRIEF (ORB) [4] are typically used for feature keypoints detection. Each algorithm is capable of detecting multiple robust features for use in vSLAM. Typically, vSLAM using these feature detection algorithms require storage of tens if not thousands of images in a database to maintain source descriptors for feature matching. With scaling becoming a major problem with larger database sizes, considerations towards reducing the storage size and recall time of the database are beneficial. Some of these algorithms can take seconds to perform with more than a few tens of images.

Fuentes-Pacheco et al. [5] discussed the problem of dynamic environments in SLAM. The authors mentioned the importance of having reliable algorithms with appreciable performance under conditions like variable light, occlusions, featureless regions and other unforeseen situation. Intensity variations are problematic indoors as there are many sources of light that can project uneven light intensities on the captured scene. If a feature detector cannot work well under varying lighting conditions then mismatches will occur and the resulting pose error will be high.

Variation in camera pose is an issue in indoor robotics as images can be taken from about any direction while navigating in small enclosed spaces. The problem is magnified when taking images from a heterogeneous mix of cameras with different fields of views and calibrations. This is especially the case as shown below in Fig. 1 with two camera sources. An example indoor system for deploying vSLAM is depicted in Fig. 1 where an Unmanned Aerial Vehicle (UAV) is cooperating with an Unmanned Ground Vehicle (UGV).

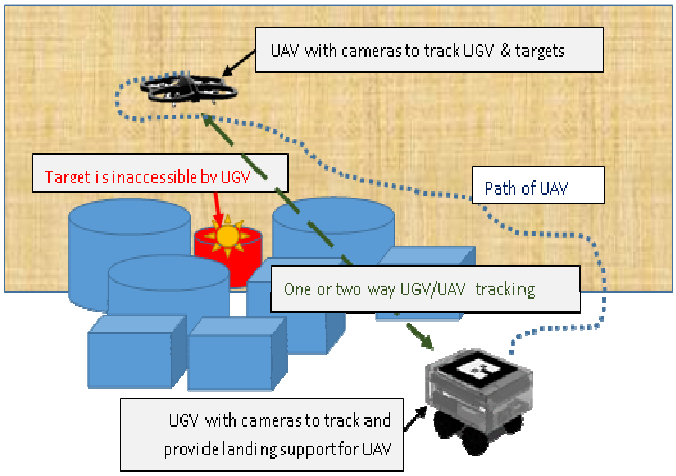


Fig. 1. Indoor cooperative robots to utilize realtime vSLAM

With image information coming from multiple poses within the environment from both camera sources, as in Fig. 1, lighting and pose variation are problematic.

Tradeoffs and sacrifices are made to achieve processing of sufficient environmental data for a limited range of working conditions. For example, resolution of processed sensory data will suffer if the processing speed is made a higher priority.

Data compression or dimensionality reduction methods are typically employed to increase processing speed.

The feature matching operation in vSLAM can be broken down into one of a simple database matching problem, where the database can be split across multiple locations. Typical robot configurations employ one computer, or multiple computers networked together either on the robot or in a local network. With a supply of computing resources available, parallelization of the database search can either occur at a processor or network level. Clipp et al. in [6] presented a multi-process CPU and GPU-based system for performing SLAM. Their system exploited the computing power within the video card to perform operations on images.

In [7], Hunziker et al. presented a real-time cloud-based implementation of SLAM running in the cloud. Much emphasis in their work, however, was not to show the performance of SLAM but the topology of their configuration that enabled the parallelization.

In this paper we aim to improve the qualities of vSLAM for use in a real-time implementation on a robot. The contribution of this paper are the algorithms to improve database storage size and reduce match recall time via feature rich database image cropping, and the experimental results into the effectiveness of our algorithm under typical indoor conditions.

The rest of this paper is organized as follows. Section 2 presents our survey of existing approaches. Section 3 covers the proposed algorithm and computing model. Section 4 presents comparative results for our algorithm versus existing well-known algorithms. Conclusions are presented in Section 5.

II. Proposed Algorithm and Computation Model

A. *Best Feature Cropping using Intensity Images*

Image intensity features are obtained via a well-known ratio algorithm known as rg-Chromaticity. In rg-Chromaticity, RGB image channels are examined on a pixel-by-pixel basis comparing the relative intensity of a single channel to the total pixel [8]. The obtained ratio is more resistant to light intensity changes compared to processing RGB channels for color information. Therefore, we propose the use of rg-Chromaticity for use in feature detection as high concentrations of red, green and blue can be more easily distinguished. Higher concentrations may yield features that are more unique in a certain scene.

Images undergo a process detailed in Fig. 2 to be added to the database used in feature matching. Images are first converted to the three channels in rg-Chromaticity which are processed for the largest concentrations of relative red, green and blues color intensities. Upper and lower x/y bounds encompassing the largest of concentrations are used as a region of interest (ROI) for cropping the original RGB image. The resulting cropped images are then inputted to the FAST feature detection algorithm. With the number of features determined for each cropped RGB image, a minimum threshold is applied and the image generated from the winning rg-Chromaticity channel is selected. If all processed cropped images are not selected, then the entire original RGB image is stored in the database. For channels that meet the threshold, the ROI cropped image is then stored in the database.

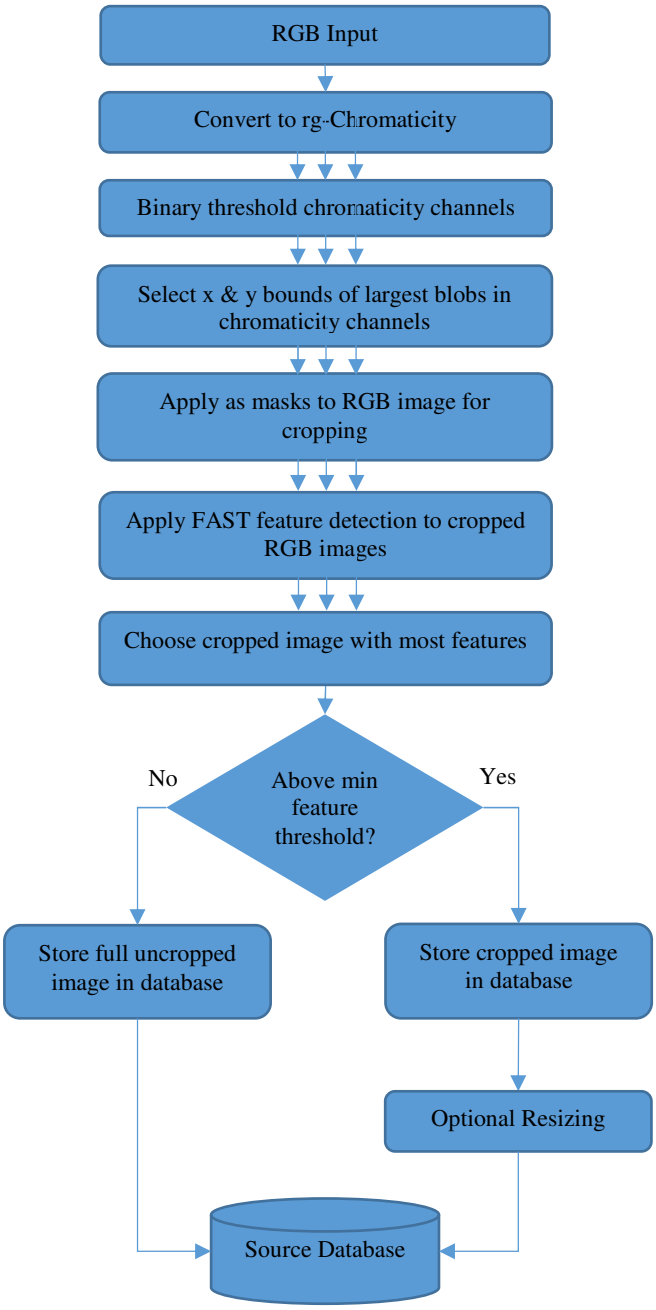


Fig. 2. Feature extraction method using rg-chromaticity rich feature areas

B. *Feature Matching Method*

Feature matching is performed using standard methods detailed in Fig. 3. Images are taken from a database or image stream in an RGB format, passed to a feature matching algorithm that uses a source image database for comparison and outputs a frame that is used for relating back to a map.

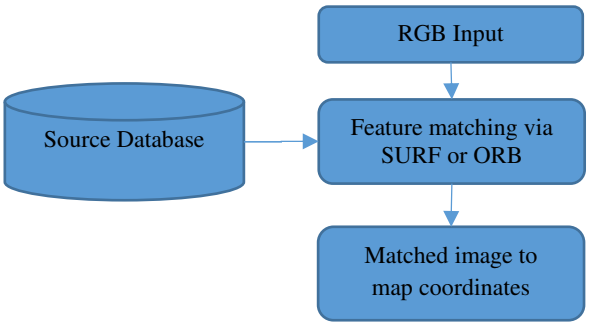


Fig. 3. Feature matching method using RGB

C. *Parallel Computation Model and Software*

The Robot Operating System (ROS) [9] is an open-source middleware for robotics that promotes robot software design as that of multiple networked processes, or nodes, that each perform a single task. With individual tasks spread out across several nodes, the task of programming of a robot is to be less challenging due to reduced software debugging time. A special node in ROS called a "master node" orchestrates communication between individual nodes and their data storage. Nodes can reside on different physical machines based on the desired topology of the processing network. A parallel vSLAM system is shown below in Fig. 4 which is designed conceptually as a set of ROS nodes.

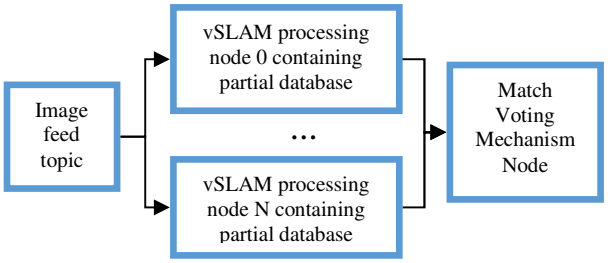


Fig. 4. Parallel vSLAM system extended with ROS concepts

To explicitly exploit multiple processes with ROS on a single Ubuntu Linux computer, the CPU affinity of each node must be set to reflect the desired operation of assigning a node to a particular processor.

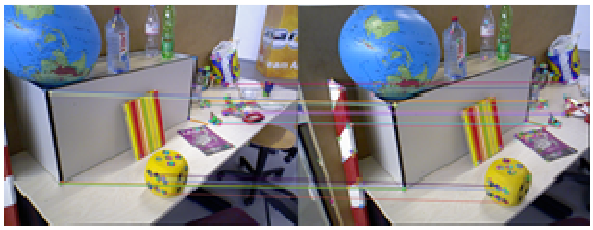
III. Experimental Results

In this section we describe results from our algorithms on cropping and resizing. All experiments are performed in python with OpenCV [10] and utilizing only a single core of an Intel Xeon 5140 Dual Core 2.33GHz CPU with 2GB of RAM.

A. *Datasets*

Image datasets are obtained from a readily available RGB-D image database source [11] for the purpose of testing RGB SLAM techniques. Furthermore, the dataset has both raw images and ROS compatible "bag files" for playback within the software developed for robotic applications. Two datasets for each general scene are provided in the database. One is a training sequence and the other a validation sequence. Each set of training images tested are fed into the feature extraction methods for generation of the vSLAM database. Feature matching is tested against a random selection of 50 images from the validation database. The random selection of validation images are the same across each tested algorithm.

Datasets **fr3/long_office_household** and **fr2/pioneer_360** from [11] were chosen for creation of the feature database. Corresponding validation datasets **fr3/long_office_household_validation** and **fr2/pioneer_360_validation** were used for feature matching against the stored database images. These datasets were specifically chosen for the variation between training and validation images in both intensity and point of view to test the proposed methods. Example images from the datasets are displayed below in Fig. 5.



(a) Database image (b) Evaluation image

Fig. 5. Example output from ORB matching in **fr3/long_office_household**

Lengths of datasets **fr3/long_office_household** and **fr2/pioneer_360** are 2588 frames and 1277 frames, respectively. Database matches are visually inspected and recorded in each of the following experiments as shown with a correct match above in Fig. 5. A total of 50 images are chosen at random from the evaluation datasets and are used for each of the experiments.

B. *Cropped RGB Images with rg-Chromaticity Feature Rich ROIs*

Results for tests of the proposed algorithm are shown in this section. Fig. 6 shows an overall successful database match with a small number of feature mismatches.

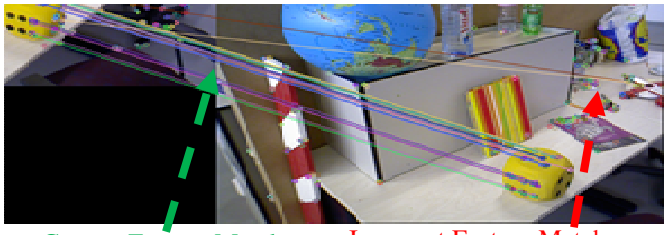
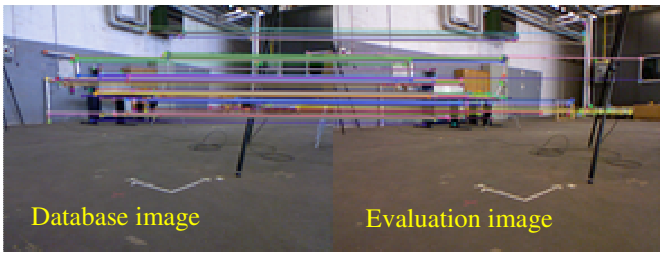


Fig. 6. Match result using database **fr3/long_office_household**.

Matching output is shown below in Fig. 7 where cropping resulted in nearly a 75 percent data size reduction for this particular image. The cropped database image in Fig. 7 (b) contains most of the features found on the uncropped image matching (a). Missed features from Fig. 7 are shown in Fig. 8.



(a) Full database image matching

(b) Chromaticity rich feature cropped image matching

Fig. 7. Matching using ORB with (a) uncropped images and (b) cropped images using proposed algorithm on database **fr2/pioneer_360.**

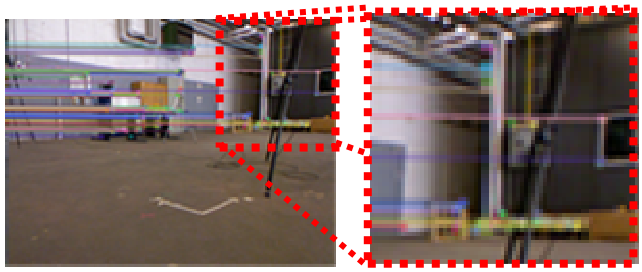


Fig. 8. Missed feature matches in Fig. 7 (b) due to cropping.

Results for the cropping operation are shown in Fig. 9 and Fig. 10.

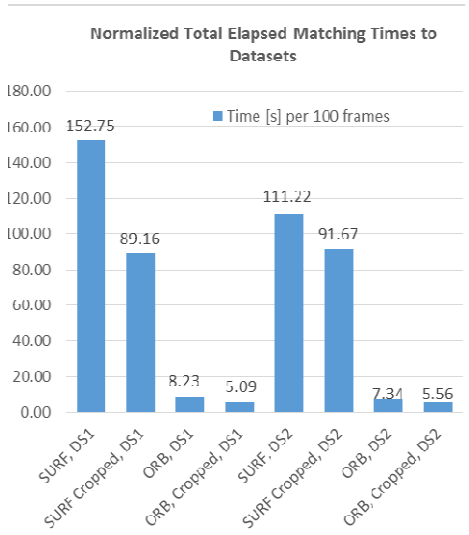


Fig. 9. Normalized elapsed times matching 50 validation images to two datasets. Note: DS1 is **fr3/long_office_household** and DS2 is **fr2/pioneer_360**.

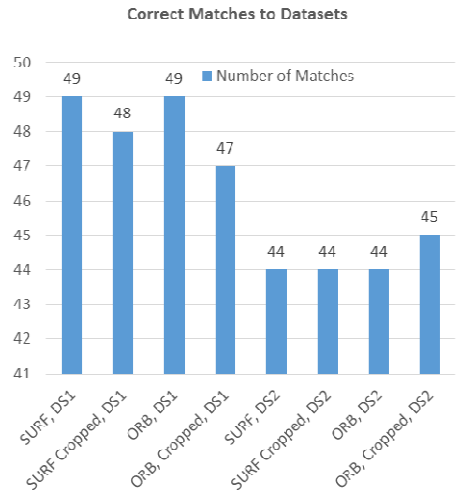


Fig. 10. Numbers of correct matches of 50 validation images to two datasets. Note: DS1 is **fr3/long_office_household** and DS2 is **fr2/pioneer_360**.

Both SURF and ORB combined with our proposed algorithm provides a significant decrease in required matching time. From the results, it is clear that SURF is not sufficient for real-time operations. SURF can take up to 18.56x more time to complete than ORB without modification.

C. Size Reduction of Datasets

Database size is dependent only upon FAST and our proposed algorithm. A breakdown of the resulting database sizes is provided in this section. Images are stored in the database using PNG file format. In Fig. 11, there is up to a 94.8% reduction in dataset size using our cropping method and the highest level of resizing (1/16 of original size) performed on the dataset.

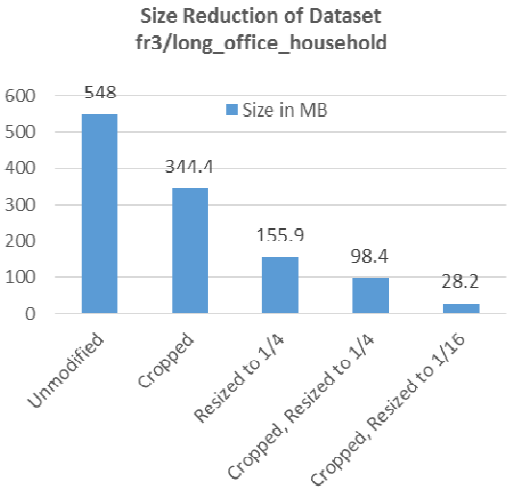


Fig. 11. Database size reduction with dataset **fr3/long_office_household**

Fig. 12 shows up to a 97.5% reduction in dataset size using our proposed method of feature-rich cropping and the highest level of resizing (1/16 of original size by area) performed on the dataset.

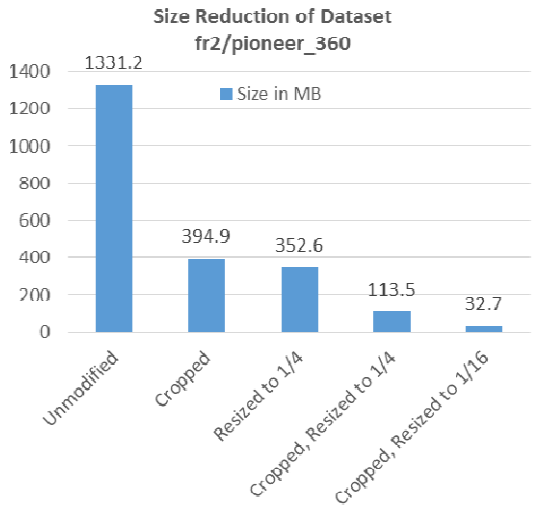


Fig. 12. Database size reduction with dataset **fr2/pioneer_360**.

D. Combination of Resized Images and Feature Rich ROIs in RGB Database

Resized images in the database are known to reduce the database size and match time [4]. Here we examine the effects of reducing image size on correctness of matching, database size and matching time to compare to our proposed algorithm. With the effectiveness of both feature rich cropping and resizing of database images, the combination of the two algorithms is also examined in this section with the datasets. Fig. 13 shows the total matching time normalized to dataset size.

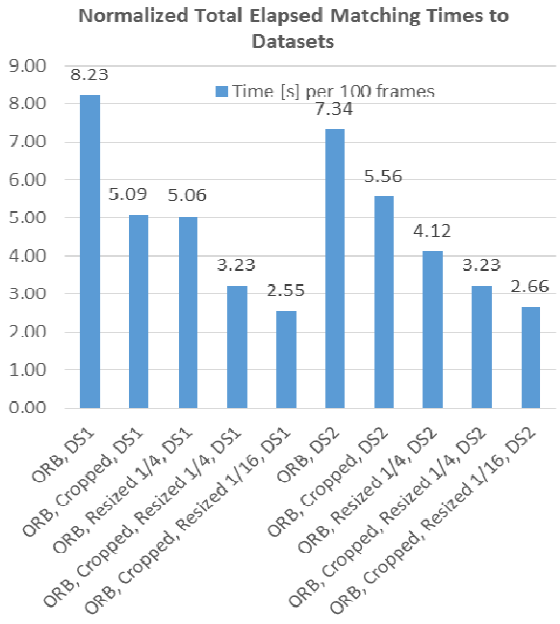


Fig. 13. Elapsed times matching of 50 images from validation datasets to stored datasets. Note: DS1 is **fr3/long_office_household** and DS2 is **fr2/pioneer_360**

Fig. 14 shows the number of correct matches between the database dataset and the validation dataset. One interesting thing to note is that the matching performance degraded at resizing down to 1/16th the original cropped image size with the dataset **fr3/long_office_household**. Only 54% matching was achieved compared to the 90-98% the algorithms achieved otherwise. Examining the dataset, one item that appeared to degrade the results is shown in Fig. 15.

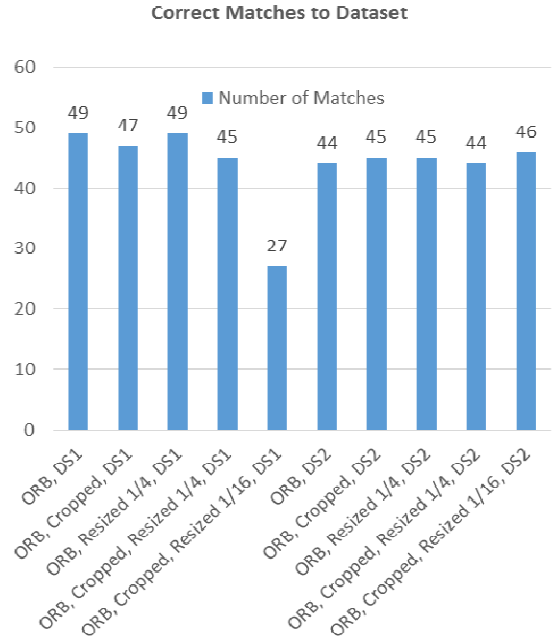


Fig. 14. Numbers of correct matches with 50 validation images to stored datasets. Note: DS1 is **fr3/long_office_household** and DS2 is **fr2/pioneer_360**

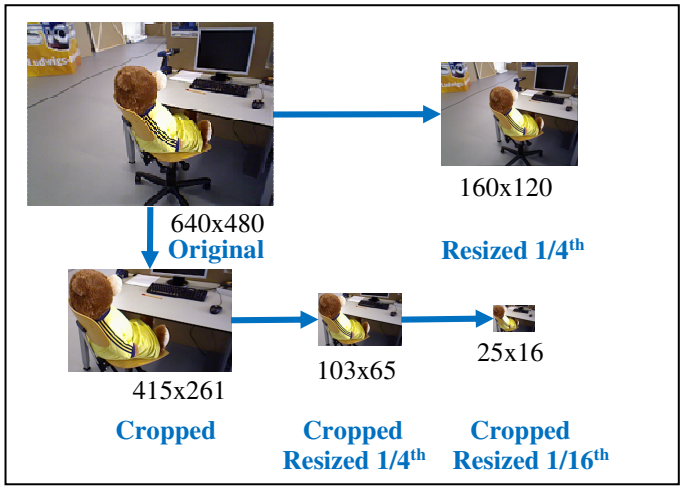


Fig. 15. Demonstration of the method utilized for resizing and cropping for database storage. Note the level of detail at an image size of 25x16 pixels.

With the level of detail provided by the landmark item in Fig. 15 at a cropped and resized level of 1/16th, the possible cause is a lack of detail on a large color-rich landmark.

IV. Conclusion and Future Research

Database storage and recall time reduction were accomplished by using smaller, more feature rich database images with selection criteria using rg-Chromaticity. In the case of dataset **fr2/pioneer_360**, use of the feature rich portions of database images yielded more matches between during validation. This effect is likely due to the cropped-out portion normally causing an incorrect match to other database images. In the case of dataset **fr3/office_long_household**, the case was the opposite with two less matches in the case of ORB and one less match with SURF. Comparing the two datasets qualitatively, **fr2/pioneer_360** contains less colorful objects while **fr3/office_long_household** has more significantly color rich scenes. It is shown that with the proposed algorithm, database size can be reduced by as much as nearly two orders of magnitude without much change in matching accuracy. Even with minimal resizing ($1/4^{th}$ of original cropped image size by area) and applying the proposed algorithm, the storage gains are significant when compared to the fact that little important data is lost, if any, as shown in the experiments. Proceeding to resize images further beyond $1/4^{th}$ of the original size provides some gains but may not be advised.

In order to further reduce recall time, use of spatial locality properties in database images is proposed to limit the search space to localized clusters. With localized clusters, a database search using feature comparisons can be greatly reduced in scope. One problem foreseen with the localized cluster approach is in the transitions between clusters. In transitions, two or more clusters may need to be checked for the exact direction of travel by the robot to correctly switch between regions.

References

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision,* vol. 60, pp. 91-110, 2004.

[2] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision–ECCV 2006*, ed: Springer, 2006, pp. 404-417.

[3] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Computer Vision–ECCV 2006*, ed: Springer, 2006, pp. 430-443.

[4] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011, pp. 2564-2571.

[5] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: a survey," *Artificial Intelligence Review,* pp. 1-27, 2012.

[6] B. Clipp, L. Jongwoo, J. M. Frahm, and M. Pollefeys, "Parallel, real-time visual SLAM," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 2010, pp. 3961-3968.

[7] D. Hunziker, M. Gajamohan, M. Waibel, and R. D'Andrea. (2014, 2014). Rapyuta: The RoboEarth Cloud Engine - RCE2013.pdf. Available: http://roboearth.org/uploads/RCE2013.pdf

[8] C. Balkenius and B. Johansson, "Finding colored objects in a scene," *LUCS Minor,* vol. 12, 2007.

[9] WillowGarage. (October). *Documentation - Robot Operating System*. Available: http://www.ros.org/wiki/

[10] A. Mordvintsev and A. K, "OpenCV-Python Tutorials," 2014.

[11] L. L. Kevin, Bo; Xiaofeng, Ren. (2013). *RGB-D (Kinect) Object Dataset*. Available: http://www.cs.washington.edu/rgbd-dataset/