A COMPUTER VISION SYSTEM FOR SPACEBORNE SAFETY MONITORING

Faisal Qureshi[1], Diego Macrini[1], Desmond Chung[2], James Maclean[2], Sven Dickinson[1], and Piotr Jasiobedzki[3]

1. Dept. of Computer Science, Univ. of Toronto, Toronto, Canada.

2. Dept. of Electrical and Computer Eng., Univ. of Toronto, Toronto, Canada.

3. MDRobotics Ltd., Brampton, Canada.

ABSTRACT

We propose a computer vision system, called a "visual supervisor", capable of visually monitoring an environment with the goal of safety monitoring and task verification. We demonstrate our system in a spaceborne setting in which from monocular video sequences, it 1) tracks and recognizes objects, 2) detects workspace violations, and 3) supervises in-progress tasks for anomalous situations, failures, or satisfactory progress. The system combines motion segmentation, object tracking, object identification, and pose estimation with cognitive reasoning to construct qualitative interpretations of the scene, such as, "An object, believed to be a satellite, appears to be on a collision course with a second object, believed to be the Space Shuttle." We demonstrate our framework on the footage of mission STS-87 as viewed from the Space Shuttle Columbia, where the proposed framework correctly identified that the mission had failed.

Key words: Qualitative/Cognitive vision;Task monitoring.

1. INTRODUCTION

There are a number of space-based robotics tasks for which computer vision can play a critical role. One such task, critical to the Canadian Space Agency's role in the International Space Station, involves the automatic grasping of a known object by a robotic arm. A calibrated vision system can provide the instantaneous position and orientation of the object with respect to the camera, allowing the robot to visually track and thereby grasp the object. Although there exist techniques in the computer vision community for computing (and tracking) the pose of a geometric object from a 2-D image (sequence), these techniques can fail under extreme lighting conditions, such as those found in a space-based environment. Unfortunately, such failures cannot always be readily detected by the tracking module. Furthermore, even when a failure can be self-detected, automatic recovery (pose correction) may not be possible without manual intervention. These task-specific modules overconstrain the world, which is both a blessing and a curse. For when the environment changes unexpectedly, their myopic view of the world can render them helpless.

To cope with this limitation, we introduce the concept of a "visual supervisor" module that can visually monitor the environment with the goal of safety monitoring and task verification. The visual supervisor can track moving objects, label the objects according to prototypical classes, detect and react to events in a larger context, and may provide initialization and operational constraints to the task specific modules. The resulting framework therefore consists of a hierarchy of visual behaviours, each sensing the world at a different level of resolution. Task-specific behaviours, such as pose estimation, tracking, and grasping subsume lower-level behaviours, such as camera control and obstacle avoidance. The task-specific behaviours, in turn, are subsumed by the visual supervisor, which can both monitor and govern their performance as well as their interaction with other task-specific behaviours.

One of the highlights of the proposed scheme is that it brings together qualitative and quantitative visual processes in a behavior-based control environment. This coupling is motivated by the fact that although task-specific vision, such as CAD-based vision and tracking (Ref. 1), can yield accurate metric pose information, it cannot accommodate abnormal conditions, such as extreme illumination changes or unexpected objects. A more qualitative visual "front end" that can identify and track the qualitative shapes of multiple objects without knowing their exact geometry can provide a level of robustness essential for space robotics.

Operations in space have evolved for the most part with a human presence. As the capabilities of robotic systems improve, they offer a viable way to reduce the costs and risks associated with human presence in space. Ground controlled, semi- and fully autonomous space systems will require complex vision systems that can operate un-

der space illumination, as well as a large range of distances and viewpoints. Current space vision systems are designed for single functions only and rely on operator assistance. Future vision systems will incorporate different strategies (modules), use different modalities, monitor their operation, and function autonomously (Ref. 2). Autonomy is the key requirement for future space vision/robotics applications, such as those listed below:

- Robotic applications for reuseable space vehicles
- On-orbit satellite servicing
- Planetary exploration
- Servicing of scientific payloads
- Support to space infrastructure development
- Utilization and maintenance of existing and future space infrastructure

Furthermore, as reliance on autonomous controllers increases, visual supervisors, such as the one described in this paper, become increasingly relevant. The visual supervisor will not only monitor the task-specific visual processes, but also monitor the *interaction* of these processes. For a robot mounted on the space station to grasp an object, for example, one might envision a visual supervisor whose field of view not only contains the robot's workspace but also the surrounding region. From this vantage, the visual supervisor can not only examine the progress of the task involving the robot, but also detect if an unauthorized object is about to encroach on its workspace. The visual supervisor thus provides a visual "sanity check" on what's happening in the environment.

We demonstrate our visual supervisor framework on the footage of mission STS-87— an actual satellite acquisition task—as viewed from the Space Shuttle Columbia. The system correctly observes and reports the anomalous condition that the acquisition has failed.

The paper is organized as follows. We begin by presenting the system overview in Section 2. Then in Section 3, we explain the motion segmentation and tracking module. Section 4 describes the object representation and matching scheme. In Section 5, we introduce our spatio-temporal reasoning scheme for object identification. Section 6 introduces task hypothesis generation and verification scheme. We conclude the paper with results and conclusions in sections 7 and 8, respectively.

2. SYSTEM OVERVIEW

The visual supervisor follows a hierarchical visual processing paradigm where low-level visual routines, such as motion segmentation and tracking, support higher-level processing that involves spatio-temporal reasoning (Figure 1). A long-standing challenge in computer vision

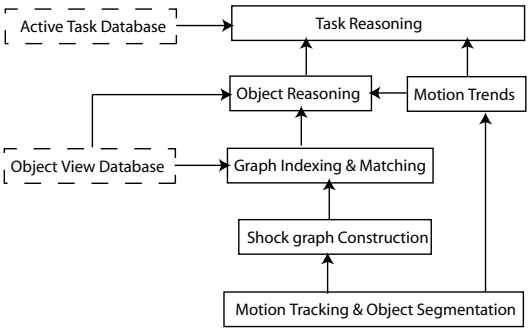


Figure 1. The Visual Supervisor: An Overview

is to successfully combine low-level visual routines with knowledge-based reasoning to construct high-level qualitative interpretations of a scene that go beyond the usual interpretations, such as background-forground separation or motion tracking. We address this challenge by employing CoCo—an agent control framework that combines reactivity and deliberation by managing the cogitation-action cycle (Ref. 3).

Due to extreme lighting variation in a space environment, appearance-based recognition modules are ineffective for object identification. The only stable feature that we can exploit for object identification and pose estimation is the shape of an object's silhouette. Our framework therefore begins by recovering the silhouette boundary of a moving object, and then recognizes the object based on the qualitative shape of the silhouette. We will describe these two modules in greater detail in the following sections.

The first step involves tracking regions of coherent motion, corresponding to multiple, independently moving objects, against a possibly moving background, in the field of view. We then construct a part-based representation consisting of a collection of qualitatively defined parts, called a shock graph, from the bounding contour of a given coherently moving region. Next, an integrated graph indexing/matching module matches the recovered shock graph against a database of shock graph "views" of known objects, resulting in a ranked list of object-view hypotheses.

The reasoning module uses the available motion history and object pose hypotheses and employs weak, domain-independent notions of spatial and temporal coherence to quickly identify objects and characterize their motions. Given *a* priori knowledge of the currently active tasks, the system attempts to compare the observed object behavior with the expected object behavior. Unrecognizable objects or unexpected object motions can be used to issue warnings and invoke task specific routines to further investigate the anomalies.

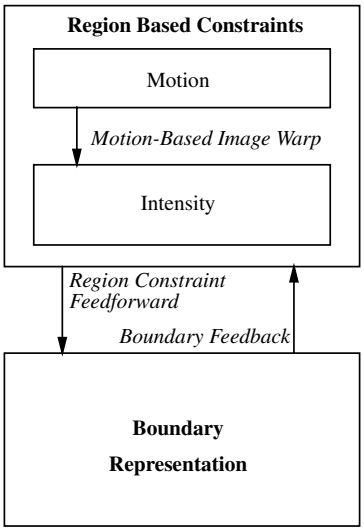


Figure 2. Region-based information is used to derive flow constraints, which provide data for the active contour. The contour, in turn, is warped between frames according to the motion parameters. The contour reinforces spatial coherence by considering only those motion constraints that lie within the contour.

3. MOTION SEGMENTATION AND OBJECT TRACKING

The tracking module (Ref. 4) estimates spatially coherent motion by combining region (layered motion estimation) and boundary information (active contour estimation). The tracking scheme employed here assumes no *a priori* knowledge about the number of regions to be estimated or the shape of the regions. Furthermore, it does not assume a static background or a stationary camera. These attributes make the motion estimation and object tracking scheme used here particularly suitable for a space environment, where color-based object tracking schemes might fail due to harsh lighting situations.

The tracking module begins by estimating a sparse set of motion constraints in successive frames by assuming brightness constancy. These motion constraints are then classified according to a constant or affine parametric model using the Expectation Maximization (EM) algorithm. Dense segmentation constraints, which are required to initialize an active contour, are generated by warping the images according to the recovered motion and evaluating the match of pixel intensities. Once initialized, the shape of the active contour is governed by both motion and image gradient information. The spatially coherent active contour influences motion estimation in subsequent frames by excluding motion constraints that do not lie within the contour, yielding a synergy between region- and boundary-based motion estimation.

4. SHAPE REPRESENTATION AND MATCHING

The object views emerging from the tracking process must be matched against a model database to determine their identity and pose. Toward this end, we represent the silhouette of each view by a shock graph. A shock graph is a shape abstraction that decomposes the skeleton of a silhouette into a set of hierarchically organized primitive parts (Ref. 5). This hierarchy is given as a directed acyclic graph (DAG), in which node attributes encode local shape properties, such as the radii and relative positions of the skeleton points. The problem of comparing query views with model views is then solved by matching the query graphs against a set of candidate graphs from the model database.

If there were no noise, our problem could be formulated as a graph isomorphism problem for vertex-labeled graphs. Unfortunately, with the presence of significant noise, in the form of the addition and/or deletion of graph structure, large isomorphic subgraphs may simply not exist. We overcome this problem by proposing a matching algorithm that accounts for *local* similarity measures of graph topology and node attributes. In turn, our characterization of graph topology becomes the discriminative feature in a fast screening mechanism for pruning the database down to a tractable number of candidates. Details of the pruning algorithm can be found in (Ref. 6).

We draw on the eigenspace of a graph to characterize the topology of a DAG with a low-dimensional vector. The eigenvalues of a graph's adjacency matrix encode important structural properties of the graph, characterizing the degree distribution of its nodes. Moreover, we have shown that the magnitudes of the eigenvalues are stable with respect to minor perturbations of graph structure due to, for example, noise, segmentation error, or minor within-class structural variation (Ref. 6). For every rooted directed acyclic subgraph (i.e., a shape part) of the original DAG, we compute a function of the eigenvalues of the subgraph's antisymmetric $\{0, 1, -1\}$ node-adjacency matrix, which yields a low-dimensional *topological signature vector* (TSV) encoding of the structure of the subgraph. Details of the TSV, along with an analysis of its stability, can be found in (Ref. 6).

Each node in a graph (query or model) is assigned a TSV, which reflects the underlying structure in the subgraph rooted at that node. This encoding of graph topology allows us to discard all the edges in our two graphs and then solve the matching problem by finding the best corresponding nodes; two nodes are said to be in close correspondence if the distance between their TSVs, and the distance between their node attributes is small. Such a formulation amounts to finding the maximum cardinality, minimum weight matching in a bipartite graph spanning the two sets of nodes. We combine the above bipartite matching formulation with a greedy, best-first search in a recursive procedure to compute the corresponding nodes in two rooted DAGs which, in turn, yields an overall similarity measure that is used to rank the candidates. An

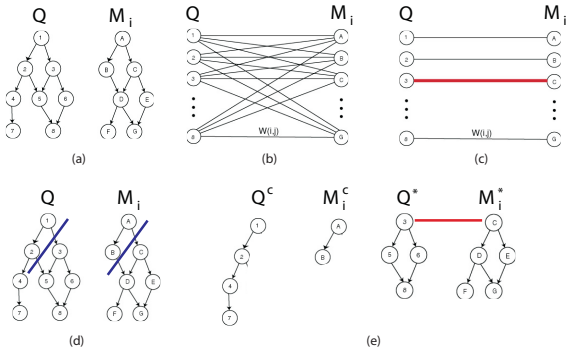


Figure 3. The DAG matching algorithm. (a) Given a query graph and a model graph, (b) form bipartite graph in which the edge weights are the pair-wise node similarities. Then, (c) compute a maximum matching and add the best edge to the solution set. Finally, (d) split the graphs at the matched nodes and (e) recursively descend.

overview of the algorithm is depicted in Figure 3, and the details can be found in (Ref. 7, 8).

5. OBJECT REASONING

A fundamental requirement for the proper function of the visual supervisor is robust object identification. The visual supervisor must be able to positively identify objects in the field of view given the shape matching results, recover from short-term segmentation/tracking failures, and handle object occlusions. To meet these requirements, the visual supervisor maintains an abstracted mental model of the scene, which is continuously updated to reflect the passage of time and the latest results from the visual processing routines. Application of spatio-temporal rules upon the mental state yields the list of objects believed to be in the field of view.

The shock graph-based shape matching module treats each frame independently and returns a ranked list of object-view hypotheses for each image region that exhibits coherent motion as estimated by the motion segmentation and object tracking routines (Figure 4). Depending upon the quality of the bounding contour and the number of objects in the object-view database, the hypotheses might be ranked incorrectly, or worse, do not even include the target object. What further compounds the problem of choosing a candidate object for the region is the fact that object-pose hypotheses for a given region of motion can vary considerably between frames. The object reasoning module accounts for the noisy nature of motion segmentation and object tracking by maintaining a small, manageable number of candidate objects, instead of committing to a single object for a given region.

The object reasoning module examines object-view hypotheses across multiple frames to select candidate objects for a given region. It begins by accumulating votes for each hypothesized object for a given region over multiple frames. Objects that occur more frequently receive higher votes, and those above a certain threshold are kept as possible candidates for the region. The success of our voting scheme depends on the assumption that inaccuracies in object hypotheses are temporally inconsistent and unstable, i.e., a given region will not be assigned the same incorrect object across multiple frames. We observed empirically that the above assumption holds true in our setting.

INDEXING RESULTS

Rank	Object Name	View #	Similarity
0	fuzzy_spartan	315	1
1	fuzzy_spartan	225	0.868278
2	fuzzy_spartan	270	0.868278
3	fuzzy_spartan	135	0.868278
4	fuzzy_spartan	45	0.759339
5	BAT	16	0.546703
6	fuzzy_spartan	315	0.546703
7	fuzzy_spartan	0	0.472085
8	fuzzy_spartan	0	0.414933
9	KNIFECLV	77	0.414933
10	BAT	22	0.400738
11	BAT	81	0.400738
12	BAT	101	0.400738

MATCHING RESULTS

Rank	Object Name	View #	Similarity
0	fuzzy_spartan	45	0.753641
1	fuzzy_spartan	135	0.546489
2	dog	46	0.538645
3	pig	42	0.525449
4	fuzzy_spartan	0	0.515806
5	dog	71	0.501153
6	ESPRESSO	3	0.490926
7	camel	127	0.487582
8	camel	70	0.473799
9	dog	47	0.467701
10	pig	87	0.464735
11	fuzzy_spartan	225	0.454066
12	HORSE	86	0.449975
13	fuzzy_spartan	0	0.435783
14	camel	22	0.428474
15	fuzzy_spartan	0	0.410162

Figure 4. Object pose hypotheses with similarity value of more than 0.4 for Region 1 in the video frame. The list of hypotheses also contains objects that clearly do not exist in space, which is due to the fact that models for these objects are present in the shape matching database. The models were added intentionally to challenge the shape matching and object reasoning modules.

A naive voting scheme that involves counting the number of occurrences of a hypothesized object is not enough, as it only takes into account the frequency of occurrence without paying any attention to the pattern of occurrence (Figure 5(a)). We instead propose a weighted, occlusion sensitive, and time-discounted voting scheme, which formalizes the notion that 1) recurrent and temporally stable hypotheses are more likely, 2) objects that have not been hypothesized for the last few frames should be *forgotten*, and 3) objects believed to be occluded should retain their values at least for some time in the future (Figure 5(b)).

The number of candidates for a given region can be further reduced by employing domain specific knowledge including reasoning about object pose, invoking object specific recognition routines, or querying a human operator. Object pose reasoning involve enumerating all possibilities within object pose space, computing a domain-dependent cost for each possibility, and keeping only possibilities that have low costs (Figure 5(c)). Reasoning about an object's pose requires guarantees about the speed at which the shape matching module operates vis-à-vis the time scale of the environment. For the current prototype, we do not assume these guarantees, so we have not yet implemented domain specific reasoning for object identification.

The object reasoning module handles object 'entry', 'exit', and 'occlusion' events by examining the number of motion regions as estimated by the motion segmentation

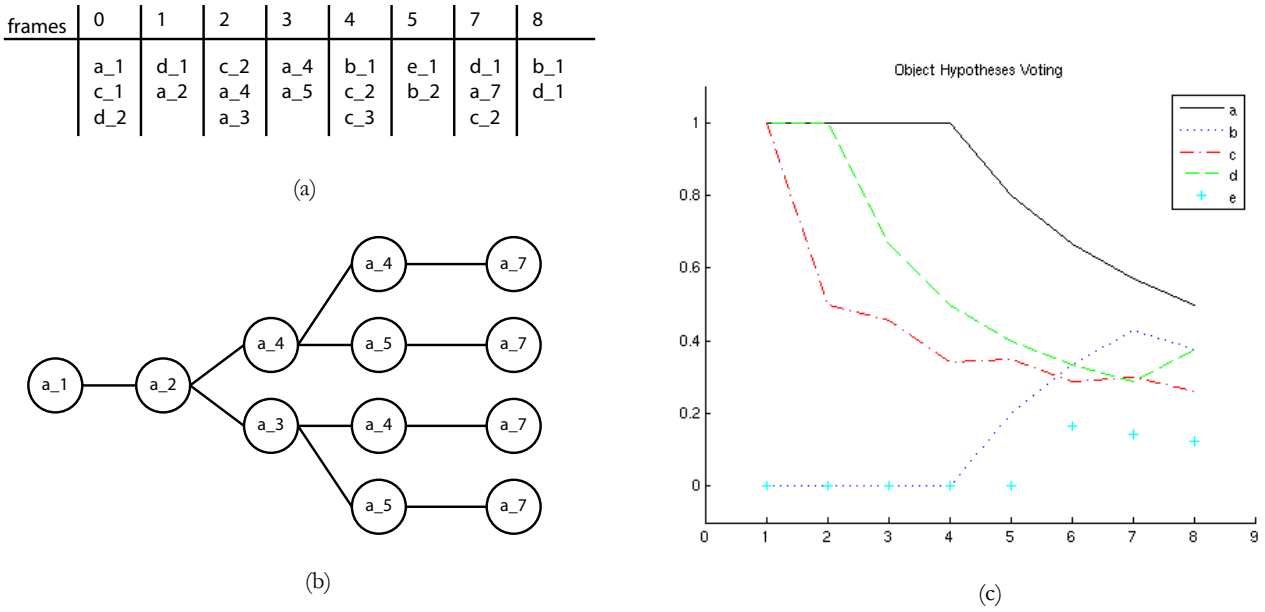


Figure 5. Object Reasoning—a hypothetical scenario. (a) Object pose hypotheses for a given region at successive frames. (c) Objects' votes accumulated over successive frames. After 8 frames, object 'a' appears the most likely candidate. (b) Object pose interpretation tree (object 'a') construction after applying an object persistence constraint. Domain specific knowledge can be used to assign a cost to every path in this tree; paths with higher costs are less likely. Reasoning within the object pose space further reduces the number of candidate objects for a given region.

routine. An 'entry' event is detected when a new region is detected through motion segmentation. The object reasoning module avoids spurious 'entry' events by enforcing spatial coherence and a temporal stability constraint on the newly detected region over successive frames, i.e., the new region must be reliably detected across multiple frames before it is considered a valid 'entry' event. An 'exit' event is detected when an object leaves the field of view, which results in a decrease in the number of estimated motion regions. The number of estimated motion regions also decreases during object occlusion, so the object reasoning module employs first- and second-order spatial knowledge to distinguish between an 'exit' and an 'occlusion' event. An 'exit' events happens when one of the regions vanishes and none of the existing regions have the same spatial coordinates. This decision is finalized over multiple frames to avoid spurious events.

The object reasoning module can also detect when two objects form a physical connection, e.g., during a successful satellite capture operation, and to which we refer as a 'coupled' event. We assume that the two objects are rigidly connected after the 'coupled' event, so we can use the number of estimated motion regions as a cue to detect a 'coupled' event. A 'coupled' event is detected when 1) the number of estimated regions of motion decreases and 2) the resulting bounding contour resembles the union of the two bounding contours, one for each object, prior to the 'coupled' event.

6. TASK VERIFICATION

Task verification is a primary requirement for safety monitoring. The idea is that the visual supervisor should not only identify the various objects in the field of view, but also establish that their interactions are safe. Here, the notion of safety is determined by the set of active tasks—the same interaction between some objects can have vastly different interpretations given the active tasks. For example, an observation that A and B are moving toward each other should be interpreted as a possible collision in the absence of an active 'A joins B' task.

Objects in the field of view determine which tasks are possible. In the absence of any knowledge about active tasks, the problem of inferring which task is active is both challenging and computationally expensive due to a large number of potential tasks that must be reasoned about; 2^n tasks, for example, are possible with n objects. The visual supervisor, therefore, restricts the reasoning about tasks to the list of active tasks, which can be regarded as a scheme for focusing attention to what is relevant.

The visual supervisor defines each task in terms of actions and actors. We can describe a task in BNF form as

```
<task> ::= <task> <stage>
<stage> ::= <stage> <action>
<action> ::= <action> <actors>
<actors> ::= <object> <actors>
```

It simply states that a task consists of multiple stages and that each stage defines the behaviors of, and interaction between, objects. This allows the visual supervisor to quickly construct a list of objects that should be in the field of view for a given task to be active. As a first step, this list is matched against the set of objects estimated to be in the field of view by the object reasoning module. The matching provides a quick way to rule out tasks that are not active and to identify objects that should not be present in the field of view.

The matching is non-trivial due to the fact that multiple objects are usually estimated for each region of coherent motion. We propose a weighted matching scheme to address this issue, where the strength of an object determines the relevance of a match. Also, different stages of a task might involve different objects, so in general, it is not sufficient to match a task-wide list of objects. The visual supervisor instead matches the object list for each stage against the objects estimated to be in the field of view. This yields a set of possible stages that might be active which, in turn, gives a hint as to what task(s) might be active.

Stage verification is performed by observing the interaction of the objects. In the current prototype, we are able to identify the following interactions between two objects:

- Two objects are maintaining their distance.
- Two objects are converging.
- Two objects are diverging.

7. RESULTS

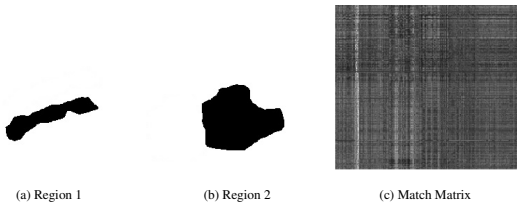
We demonstrate the framework on the footage of mission STS-87 as viewed from Space Shuttle Columbia. The objective of this mission was to capture the Spartan satellite by using a robotic arm mounted on the Space Shuttle. An astronaut maneuvered the arm to bring it closer to, and eventually dock with, the satellite. This mission was aborted when the arm collided with the satellite, sending the latter into a spin.

7.1. Stage 1: Motion Segmentation and Object Tracking

We first processed the video to identify moving objects. For this video, the motion segmentation and object tracking module correctly identified two regions of coherent motion with bounding contours shown in Red and Cyan (Figure 6).

7.2. Stage 2: Object Matching

The estimated regions are passed to the shape matching module that determines their identity and pose by matching the shock graph representation of a region's bounding contour against a model database (Figure 7). Our model database contains multiple views of 20 different objects, including 22 views of the Spartan satellite and 4 views of the robotic arm shown in the video. The performance of the shape matching module, given the fact that the model database contains 633 unique views, is a testament to the robustness of our shape matching scheme.



(a) Region 1 (b) Region 2 (c) Match Matrix

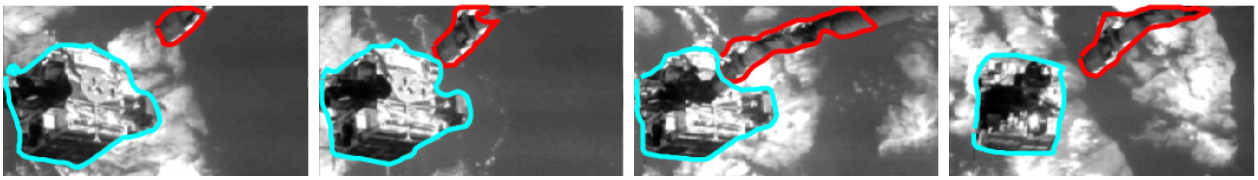
Figure 7. The estimated regions, such as those for Frame 855 shown here (a)-(b), are passed on to the object matching module that returns a ranked list of object-pose hypothesis for each region. For Frame 855, the top five matches for Region 1 are 'satellite', 'arm', 'arm', 'guitar', and 'guitar', and for Region 2: 'umbrella', 'satellite', 'satellite', 'umbrella', and 'satellite'. The correct match for Region 1 is 'arm' and for Region 2 is 'satellite'. (c) The results of matching the estimated regions in every frame against the model database. Rows represent the regions in every frame (a total of 772 rows; 2 regions in 386 frames; we processed every fifth frame starting from Frame 25). Columns (633) represent the number of views in the model database. White indicates a high-similarity value.

We created view-based models for the satellite (to be used by the shape matching routine) by rendering silhouette images of the 3D satellite CAD model from multiple views (Figure 8(a)) and then computing shock graphs from these images. It turns out that the 3D satellite CAD model has minor structural detail not visible on the actual object, which results in a shock graph that is noticeably different from a shock graph that is extracted from the bounding contour of the region corresponding to the actual satellite. This translates into poor performance of the shape matching routine. We resolve this issue by applying a Gaussian filter on the silhouette image, thereby bridging the representational gap between model and image.

We also created view-based models for the robotic arm shown in the test video. Here, due to lack of a readily available geometric model of the arm and given the fact that it is never fully visible in the video, we extracted a silhouette for the arm for every frame in the video, and clustered them into 4 exemplar silhouettes through K-Means clustering. We then computed shock graphs for these exemplars and added these to the model database.

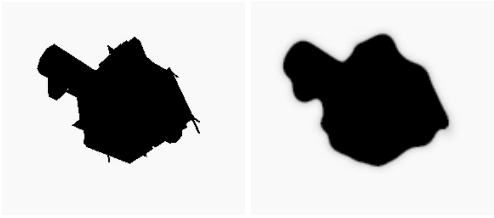
7.3. Stage 3: Object Reasoning

The object reasoning module identifies the two regions from the object-view hypotheses returned from the shape



(a) Frame 25 (b) Frame 300 (c) Frame 855 (d) Frame 1890

Figure 6. Motion tracking and segmentation routine successfully estimated two regions exhibiting coherent motion and computed their bounding contours (a)-(d).



(a) (b)

Figure 8. (a) Silhouette image by rendering a 3D satellite CAD model; notice the fine detail. (b) The same image after Gaussian smoothing.

matching module . It only considers hypotheses whose similarity value exceeds 0.4. The object reasoning module has correctly estimated Region 1 to be the satellite and Region 2 to be the robotic arm (Figure 9).

7.4. Stage 4: Task Identification and Verification

In the current setup, the set of active tasks contains only one task: 'dock'. During a 'dock' task, a robotic arm moves closer to the satellite until the latter is successfully captured. Upon a successful capture the satellite and the arm are rigidly coupled and exhibit the same motion characteristics. Given this description of the 'dock' task, we can describe it as follows:

```
<dock> ::= <approach> <captured>
<approach> ::= <converge> <sat> <arm>
<captured> ::= <coupled> <sat_arm> |
               <coupled> <sat> <arm>
```

The visual supervisor matches the objects identified in Stage 3 with the actors (or objects) required for the various stages of the active task and decided that both objects are expected to be in the field of view. In a more general setting when one of the objects is not expected to be in the field of view, the visual supervisor can either raise an alert or employ a more suitable routine for further visual analysis.

The next step is to resolve the current stage of the task. In general, specialized routines are required to decide between different stages; however, we were able to use the relative motion of the two objects to choose the active stage. We label the relative motion as converging, diverging, and coupled by fitting a linear model to the tracking data and examining its slope. Two objects that appear to be converging in a video do not necessarily imply that these are also coming closer in 3D, so the visual supervisor also estimated the relative depth of the two objects by comparing their image area to the model dimensions. Monocular depth estimation, while not as accurate as stereo depth estimation, is sufficient for our purpose. Also, the visual supervisor can potentially employ a better depth estimation strategy when the need arises.

The visual supervisor estimated the current stage to be 'approach', as the two objects exhibited distinct motions. The visual supervisor also estimated that the two objects are converging, thereby deciding that the 'approach' stage is progressing satisfactorily.

Around Frame 855 the two regions are almost touching each other, so the visual supervisor estimates that the current stage is 'captured'. Due to task failure, however, the satellite starts moving away from the robotic arm and visual supervisor detects that the two objects are diverging from each other and estimates that the 'captured' stage is failing.

8. CONCLUSIONS

We have proposed a visual supervisor framework capable of constructing qualitative interpretations of a scene from monocular videos with the aim of safety monitoring. The framework brings together motion segmentation, object tracking, view-based shape matching, and a new spatio-temporal reasoning scheme within an agent control framework. The object reasoning module can correctly identify visual events, such as 'entry', 'exit', 'occlusion', and 'coupled'. Our task description scheme allows task hypotheses to be generated from the identified objects. Low-level visual routines can then by used for task recognition and monitoring. We have demonstrated the framework on the footage from mission STS-87 and the initial results appear promising.

We envision that visual supervisors, such as the one described here, will become more relevant as an increasing number of space missions are performed autonomously by intelligent controllers without a human in the loop. As

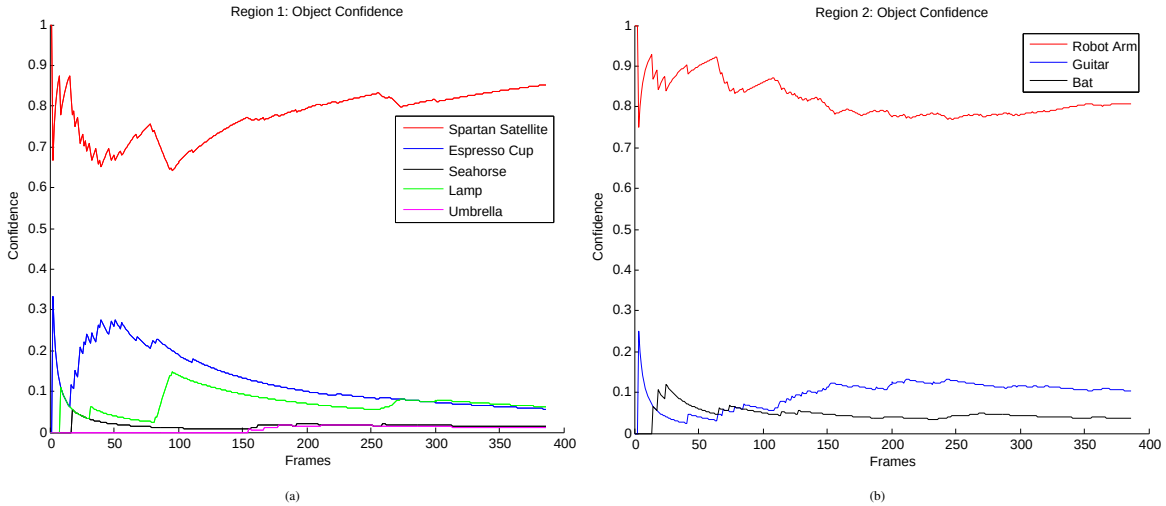


Figure 9. Voting results for possible candidates for Region 1 (a) and Region 2 (b). The object reasoning module correctly inferred Region 1 to be a satellite and Region 2 to be a robotic arm. We only show candidates whose vote at any point during the sequence exceeds 0.03.

such, the framework introduced here is a step in this direction.

There are a number of things that we want to address in the future. First, we want to evaluate the visual supervisor on monocular videos from multiple missions which, among other things, requires us to include view-based models of some other objects in the shape matching database. Next, we want to investigate more sophisticated task recognition and monitoring schemes. Most importantly, we want to close the loop between the reasoning module and the low-level visual routines, which would allow the reasoning module to guide the low-level visual routines, making them more competent and robust.

ACKNOWLEDGMENTS

The authors wish to gratefully acknowledge the financial support of the National Sciences and Engineering Research Council of Canada (NSERC) and Communications and Information Technology Ontario (CITO). The authors would also like to acknowledge the earlier contribution of Neil Witcomb to the project.

REFERENCES

1. Sminchisescu, C., Metaxas, D., and Dickinson, S. Incremental model-based estimation using geometric constraints. *IEEE Transactions On Pattern Analysis and Machine Intelligence*, 27(5):727–738, july 2005.

2. Jasiobedzki, P. and Anders, C. Computer vision for space robotics: Applications, role, and performance. In *SPRO '98 I IFAC Workshop on Space Robotics, Canadian Space Agency*, pages 96–103.

3. Qureshi, F. Z., Terzopoulos, D., and Gillett, R. The cognitive controller: a hybrid, deliberative/reactive control architecture for autonomous robots. In Orchard, B., Yang, C., and Ali, M., editors, *Innovations in Applied Artificial Intelligence. 17th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert System (IEA/AIE 2004)*, volume 3029 of *Lecture notes in Artificial Intelligence*, pages 1102–1111, Ottawa, Canada, May 2004. Springer-Verlag.

4. Chung, D., Maclean, W., and Dickinson, S. Integrating region and boundary information for improved spatial coherence in object tracking. In *Workshop on Articulated and Nonrigid Motion (CVPR)*, Washington D.C., June 2004.

5. Siddiqi, K. and Kimia, B. B. A shock grammar for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, USA, 1996.

6. Shokoufandeh, A., Macrini, D., Dickinson, S., Siddiqi, K., and Zucker, S. Indexing hierarchical structures using graph spectra. *IEEE Transactions On Pattern Analysis and Machine Intelligence*, 27(7):1125–1140, july 2005.

7. Siddiqi, K., Bouix, S., Tannenbaum, A., and Zucker, S. The hamilton-jacobi skeleton. In *IEEE International Conference on Computer Vision*, pages 828–834, 1999.

8. Macrini, D. Indexing and matching for view-based 3-d object recognition using shock graphs. Master's thesis, University of Toronto, 2003.