(Draft) Invited Paper for SCI-ISAS99
Session on Intelligent Data Mining and Knowledge Discovery

Mining for Image Content

Michael C. Burl, Charless Fowlkes, Joseph Roden
Jet Propulsion Laboratory, M/S 126-347
4800 Oak Grove Drive
Pasadena, CA 91109 USA
Michael.C.Burl@jpl.nasa.gov

Abstract

This paper provides an overview of our efforts to develop algorithms and systems that are able to "mine" useful information from large image collections. One of the core capabilities targeted is the ability to find instances of specific objects in images. Such a capability would greatly enhance analysis of NASA's existing image archives and, if coupled with an autonomous agent that is able to act on the mined information, would enable new forms of scientific exploration.

Keywords: image mining, object recognition, content-based retrieval, distributed systems, database, Diamond Eye, space exploration, queries, knowledge discovery, digital libraries.

1 Introduction

There are two trends developing within JPL and NASA that have created a demand for systems that can automatically analyze images and extract semantically meaningful information. First, the amount of data returned from spacecraft missions has greatly increased due to improvements in image acquisition, communications, and storage technology, but the ability of humans to manually analyze data has remained constant. Second, it has been recognized that placing intelligent algorithms onboard a spacecraft will lead to more economical missions and enable new types of scientific exploration.

Mining useful content from image collections is very much an interdisciplinary endeavor that draws upon expertise in computer vision, image understanding, data mining, machine learning, databases, distributed/parallel computing, software design, and artificial intelligence. Unlike in traditional data mining, there is considerable difficulty involved just in determining the proper representation of the information. The standard view of a dataset as a rectangular table in which the rows are examples and the columns are measurements of particular "attributes" doesn't map well into the image mining domain. There are a number of other difficulties such as enabling users to formulate queries to look for particular objects, providing the ability to find new types of objects without reprogramming, and organizing the mined information in such a way as to support further analysis and scientific inquiry.

Despite these obstacles, there is one source of encouragement: *humans can extract meaningful information from images.* Hence, we know that a solution must exist. In contrast, in many "pure" data mining tasks it is often unclear whether useful knowledge can indeed be extracted from a mountain of abstract data. In the remainder of the paper, we discuss insights into the image mining problem that we have gained through our efforts to develop systems for analyzing large datasets of planetary imagery.

Section 2 contains a survey of potential NASA-related applications of image mining. Many of these applications have analogs in domains closer to home such as medical imaging, petroleum exploration, digital libraries and the Internet, surveillance, management of natural resources, etc. System-level issues involved in developing large-scale image mining products are discussed in Section 3. In Section 4 we describe some of the algorithms that we are using to access image content. Section 5 provides a summary and discussion of future directions.

2 Potential Applications

Systems that can automatically analyze images and extract semantically meaningful information have a number of applications both in mining large archived datasets and in onboard spacecraft scenarios. Communications bandwidth between remote spacecraft and Earth is often extremely limited. Similarly, in the ground-based setting, human analysts have "bandwidth" constraints that limit the number of images that can be examined in a day. Applying mining algorithms to extract the interesting content can provide greater *information throughput.* Coupling mining capabilities with an autonomous agent (e.g., a self-

commanding spacecraft or a steerable sensor) that can take actions based on the mined information will lead to exciting opportunities for new science and permit feedback loops to be closed onboard (especially critical in applications such as small body landing [14] where light-time delays do not permit remote control from Earth).

The Magellan mission [21], which performed detailed global mapping of 98% of the surface of Venus, provides a valuable illustration of the need for automated analysis of image content. Within this large dataset (consisting of over 30,000 images on approximately 200 CD-ROMs) there is a wealth of potential scientific information. For example, volcanism is known to be one of the dominant geological processes on Venus [25], and it has been estimated based on lower resolution data that there are on the order of one million small volcanoes in the Magellan dataset [1]. However, any comprehensive global study of the volcanoes' spatial distribution or relationship to other geological features is essentially impossible using manual analysis. Image mining techniques have been developed and applied to portions of this dataset with some degree of success [3].

Other examples abound. In cratering studies the primary quantity of interest is the size-frequency distribution of craters, which can be used to assess the relative ages of planetary and small body surfaces. Using automatic detection and sizing techniques to process the raw images directly into summary form (size-frequency distributions) can potentially provide enormous savings in labor while preserving the information of interest.

A variation on the summarization theme is "intelligent compression" in which regions around objects of interest, rather than raw images, are returned by spacecraft[1]. This idea is quite different from traditional image compression, which involves no understanding of an image's content. In some cases, intelligent compression may offer reductions in data rate of a thousand or more *with no loss in quality of the returned data.* In contrast, traditional lossless image compression offers factors of only two to three[2].

Another use of mining is to provide focusing and prioritization. For example, using a volcano recognizer and spatial reasoning, a system could guide a human user to a "short list" of areas having a high probability of containing volcano fields. The user could then focus his initial analyses on these areas. The system acts as an intelligent assistant that multiplies the human user.

Examples of image mining coupled with an autonomous agent include retargeting (imaging a detected object or event at higher resolution or with specialty sensors), monitoring (vigilantly watching an area while waiting for an event such as a volcanic eruption), tracking (maintaining focus on a specific object or event such as an atmospheric storm), and prioritizing (ordering the areas from which data is collected based on visual content).

3 System-level Issues

In developing image mining capabilities we operate from the viewpoint that any algorithms must be thoroughly tested and proven in the ground-based setting to gain acceptance and ultimately to be deployed on spacecraft missions. The testing required usually necessitates packaging the algorithms into a system that can be used and evaluated by the domain experts (collaborating planetary scientists) as part of their ongoing ground-based analyses.

When we first began assembling large-scale mining systems, we were somewhat surprised to find that only a small fraction of the code in such systems is devoted to the core algorithms. Instead, the bulk of the code provides *infrastructure*, which in broad terms includes everything that enables the user to manage and interact with data and algorithms. Since much of the infrastructure cuts across image mining applications, we have developed a distributed (web-based) system that factors out and reuses the common code. The system is called Diamond Eye as one of the core capabilities to be supported is the ability to find objects of interest, figurative "diamonds", in large image collections. Here, we briefly describe the Diamond Eye architecture and some of the system-level issues that affect the design. A more comprehensive discussion is given in [4].

3.1 Diamond Eye Architecture

As shown in Figure 1, the basic Diamond Eye architecture consists of a custom server, which is closely coupled with an object-oriented database and a computational engine. Users interact with the server through a cross-platform Java[3] applet interface. The applet allows the user to browse and annotate images, formulate queries, request data mining services, and display results. Each data mining server handles the execution of user requests and data routing between various system components. In a typical deployment,

[1] A core set of raw images will always be part of the returned data. Intelligent compression provides a way to maximize the information return from any "discretionary" bandwidth.

[2] Note that traditional compression can be applied on top of "intelligent" compression.

[3] Java is an object-oriented language developed by Sun Microsystems for platform-independent, distributed computing in a heterogeneous networked environment.

the server is coupled with a high-performance computational engine (e.g., a network of workstations) that provides parallel execution of computationally intensive image mining operations. An object-oriented database provides persistent storage to users and the system itself. The database also enables users to pose queries over the "mined" information.

3.2 Database

Because of the security restrictions imposed on applets, the Diamond Eye client program cannot access the user's local filesystem; hence, any persistent storage for information that should exist across user sessions must be provided through the server. Although the server's filesystem could be used for this purpose, a database is the preferred solution since it enables the user to easily perform arbitrary navigation and queries over both the raw data and accumulated results.

In earlier system prototypes, relational and object-relational databases were used to store the results of image mining operations. A single type of data structure, a table, is used in these databases to maintain information. The rows of each table correspond to instances and the columns correspond to attributes. Thus, a table of employees might have a row for each employee and a number of columns with information such as NAME, HOMETOWN, and HIREDATE. Since the number of different hometowns is likely to be much smaller than the number of employees, a gain in storage efficiency is possible by storing a TOWNID in the EMPLOYEE table and providing a second table that links TOWNID to the name of the town (a string). Although the relational (table) model works well for some domains, we have found it cumbersome for image mining applications. Therefore, we are now exploring the use of object-oriented databases (OODB).

A key advantage of an OODB is that it stores objects using the structures of the native programming language; the developer can think exclusively in terms of the natural data structures for the problem (in our case Java classes/objects) without mapping back and forth between the table representation and the natural representation. Further, the use of the natural data structures provides maximum flexibility in terms of the objects and relationships that can be easily represented. In particular, a number of objects that are needed in image mining such as recognition models, graphical relationships, etc. are not easy to represent in the table framework.

The OODB also exploits many of the benefits of object-oriented programming including the ability to have class hierarchies and methods (functions) associated with objects. A more complete argument for the use of OODBs in image mining is presented in [24].

3.3 Algorithm Interface

One of the fundamental constraints in the design of Diamond Eye was that it should be easy to add new algorithms to the system. This goal is achieved through the use of wrapper objects that provide a uniform way to shuttle data and parameters back and forth between the Diamond Eye system and the algorithms, which are implemented as external (stand-alone) programs and scripts. The decision to use external programs/scripts rather than function calls through the Java Native Interface (JNI) was based in part on the flexibility provided[4].

With the external program/script approach, the algorithm developer simply supplies an executable version of the code compiled from the developer's language of choice along with a Java wrapper known as an "algorithm object". All algorithm objects are derived from an *abstract* ALGORITHM class, which simply means that algorithm objects must implement a set of methods that are "promised" by the ALGORITHM class[5]. Methods that are promised include the means for the system to get algorithm-specific parameters from the user, save and load data to/from the file system (the file system serves as a form of shared memory between the system and external programs), assemble the command line string used to call the external program, properly handle the output of the algorithm (e.g., by displaying detection results on an image), and so on. More detail on the algorithm interface is given in [4].

4 Algorithms

In this section, specific algorithms that we have experimented with for extracting information from image collections are described. We note that while there is a large content-based image retrieval (CBIR) thrust within the computer vision community [10, 22, 27], the approach we use differs significantly. In the CBIR community, the focus is on retrieving images that are "visually similar". There is also an emphasis on making queries interactive so that answers are returned quickly (within a few seconds). Hence, many of the techniques are based on relatively low-level definitions of content, e.g., precomputed global properties of images such as color histograms and texture descriptors. In contrast, the Diamond Eye system is aimed at providing a "high-end" capability that can provide access to *specific objects* within images. In [11] this dis-

[4] Both approaches can co-exist; however, only the external program/script approach is currently used.

[5] Alternatively, this behavior could be implemented with a Java construct known as an *interface*.

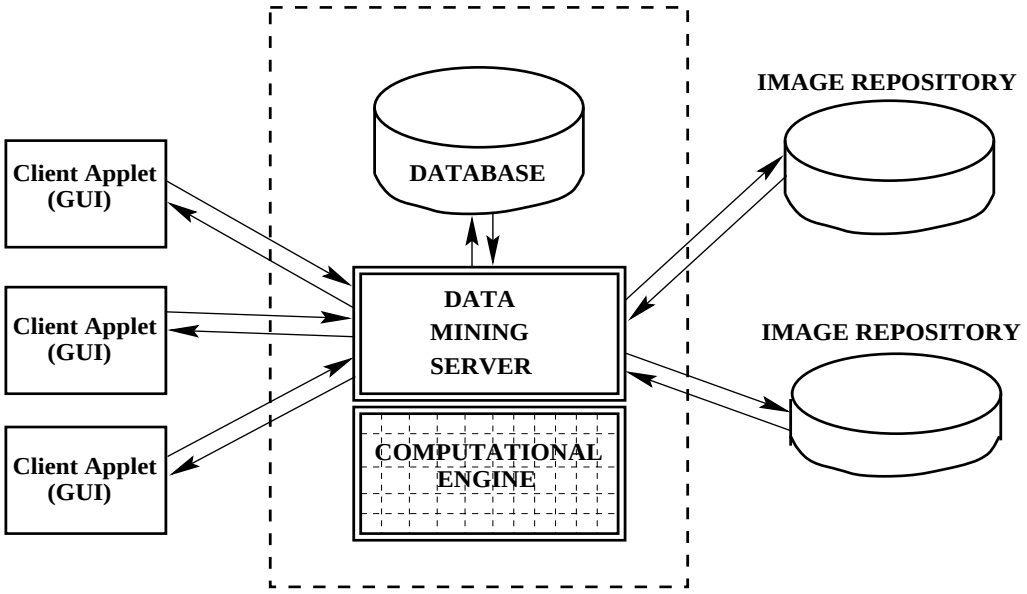


Figure 1: The basic Diamond Eye architecture consists of a custom server, which is closely coupled with an object-oriented database and a computational engine. Users interact with the server through an applet interface. The server executes user requests and handles data routing, such as retrieval of images from external databases.

tinction is described as finding "things" rather than "stuff". Due to the difference in emphasis, the Diamond Eye system employs more complex recognition models, but must rely on the availability of fast, parallel hardware to carry out its work. In subsequent sections, we present several types of recognizers and query models that are used in the Diamond Eye system.

4.1 Recognizers

A common approach for generating recognizers is to hand-code a set of routines that compute object-specific attributes from the pixel representation of an object. Often the choice of attributes is determined by interviewing domain experts to capture their knowledge as a set of high-level features and rules. There are, however, a number of drawbacks to this approach. Although the experts are quite good at identifying examples of the objects of interest, it is often difficult for them to identify precisely what characteristics of the image region caused their decision. Even when high-level features are available, e.g., circularity or symmetry, it is difficult to translate these rules into pixel-level constraints. Perhaps the biggest drawback is that a new set of attributes and rules is needed for each new object or domain.

We are pursuing a different approach in which recognizers are developed automatically through learning techniques. The knowledge of the domain expert is captured *implicitly* through a set of labeled examples identified by the expert. Learning algorithms are used to abstract an appearance model of the object from the training example(s). The model can then be used to find novel instances of the object in an image collection. This approach has the advantage that it can be used to develop recognizers for different objects without reprogramming.

4.1.1 Matched filtering models

The simplest form of recognizer has its origins in the communications and radar work of the early 1940's. From this work it is well known that the optimal detector for a known signal in additive white Guassian noise is a matched filter, i.e., a filter whose shape is exactly matched to the signal one is trying to detect. This idea has been generalized to 2-D patterns and is often referred to as template matching [9]. An object with a fixed appearance is detected by correlating an image with a filter/template that exactly matches the object's appearance. Although this approach is useful in many circumstances, it quickly collapses when

the object of interest is variable in appearance, due to (1) external variations such as the lighting, object pose, distance from the camera, etc. and (2) internal variations that occur when not all instances of the object are exactly alike (e.g., different craters or different volcanoes).

4.1.2 Continuously-scalable template models

One idea for improving upon rigid template detectors involves using the steerable-scalable-deformable filtering ideas developed by Freeman and Adelson [12], Perona [20], and others. The basic idea is that a filter with parametric dependence [19] can be approximately implemented using a finite set of basis filters. We have applied this approach to generate a scalable version of matched filtering that is useful when the primary variation in the appearance of an object is due to scale [2]. This situation occurs if the distance of the imager from the object varies or if the object naturally occurs at different sizes.

The continuously-scalable template (CST) model is generated from a single "real" example of the object, known as the prototemplate. The prototemplate is typically interactively selected from an image by the user. For less precise query-type operations or model bootstrapping when no examples are available, it is possible to "hand-draw" the prototemplate in Diamond Eye using the mouse and a palette of gray-levels.

Once the prototemplate is selected, it is spatially resampled to generate a family of templates at closely-spaced scales covering a factor of two or so in scale-space. Ideally, each member of the template family would serve as a matched filter for detecting the object at a particular scale; however, the brute-force approach of correlating each family member against an image would be too expensive computationally. Instead, the template family is compressed using singular value decomposition (SVD). This process provides a reduced-dimensionality linear basis that approximately represents each family member. The basis can be used to efficiently calculate the correlation of an image with any family member and, therefore, provides a way to simultaneously detect and size instances of the object over a continuous range of scales. Coupling this technique with a pyramid representation of the image provides continuous coverage over orders of magnitude in scale.

4.1.3 Principal components analysis models

Another form of recognizer that can be used to model variations within a class of objects is based on principal components analysis (PCA). PCA was developed by the statistician Hotelling in the 1930's [13], but the method was not applied to image recognition applications until recently [26, 3, 18]. The approach is similar to that used with the CST models. A family of examples of the target object is approximated by a reduced-dimensionality linear basis. The main distinction is that PCA is a statistical technique that requires a number of examples of the target object, whereas CST models are derived from a single example (plus artificially synthesized examples). Also, the examples provided to PCA do not depend on an underlying parameter that varies continuously.

One way to view PCA is to imagine the pixel-space examples of an object as points lying in a high-dimensional space (with dimension equal to the number of pixels in the pattern). For many vision-type problems it turns out that the examples do not lie everywhere in the space, but instead reside on a relatively low-dimensional manifold that can be approximated as a hyperplane. The basis vectors are the coordinate system of the hyperplane and the projection of an example onto the basis vectors provides a low-dimensional "feature space" characterization of the object. Machine learning techniques can be successfully applied in this feature space to discriminate object from non-object.

4.1.4 Spatially-deformable configurations

Although both PCA and CST models are useful for representing variations within an object class, these techniques suffer from the fact that they attempt to represent an object's appearance manifold *globally* with a single hyperplane. For harder problems, we believe the single hyperplane model is inadequate and have developed a new approach [15, 5, 6, 8, 7, 16] in which these local techniques are used to represent the appearance of *parts* of an object and the parts are linked together through probability distributions over their spatial arrangement. The probability distributions can be formulated in a special *shape space* that is invariant to geometrical deformations such as translation, rotation, scaling, and even affine transformations [16]. This type of model provides a useful hybrid of image-based techniques (PCA, CST) and geometry based techniques (geometric hashing [28, 23], invariants [17], etc.) The spatially-deformable configuration of parts (SDCP) models are hierarchical so that a whole object can be modeled as a spatial arrangement of parts and the parts in turn can be modeled as a spatial arrangement of subparts [7]. Like the other models discussed, SDCP models can be learned from

examples.

4.2 Queries

Queries allow a user to search through a collection of images and return the set of best matches. In the CBIR community, queries are typically based on a combination of low-level properties such as color histograms and texture descriptors. In our formulation, there is a much closer connection between recognizers and queries. In fact, the primary difference is in what the user wants for output. If a recognizer is applied to a collection of images, the user most likely wants to generate a complete catalog of *all* instances of the target object. With queries, the user's focus is instead on finding *some* instances. Hence, query results are presented as a set of thumbnails showing the set of best matches. A second implication is that query models do not need to be as precise as recognizer models. Thus, queries often are constructed through a visual interface rather than via statistical techniques like PCA. For example, a visual query-specification interface included in Diamond Eye enables an SDCP query to be generated based on a single example of the object of interest and relevance feedback from the user.

4.3 Libraries of Recognizers and Queries

As reliable models are developed for recognizing specific objects, the models can be stored in the system database and tagged with a name according to the type of target object. Figure 2 shows a cartoon representation of several recognizer/query models. A user, who is preparing to search for some type of object, such as craters, can first check the database to determine if such a model already exists. As the user base grows and more models are contributed to the database, users will become able search for many different types of object simply by specifying the name of the object. The system will pull the appropriate recognizer/query model from the database and apply the model to one or more images as requested by the user. In the future, users will have the ability to rate models along different dimensions (speed, accuracy, etc.) and provide feedback that can be used to refine a model (e.g., the following identifications produced by crater model *xyz* do not appear to be craters).

5 Summary

Automatically mining or extracting useful information from an image collection is a difficult problem that requires expertise from multiple disciplines. However, the potential benefits are compelling, especially when image mining capabilities are coupled with an autonomous agent that is capable of acting upon the extracted information.

The work we have described has largely focused on the case where a model of the object of interest is developed based on previously seen examples. As spacecraft and robotic platforms explore new environments such as Pluto and the subsurface of Europa, we will not always know what to expect in advance. For example, the Voyager fly-by of Neptune's moon, Triton, in the 1980's revealed the existence of "ice geysers", which were a new type of geologic feature that had not been seen elsewhere in the solar system. The development of a *discovery capability* to identify such content in images without a strong prior model for the appearance provides an interesting direction for future work.

Acknowledgments

The research described in this article has been carried out in part by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. One of the authors (C.F.) participated in this project under the umbrella of the Summer Undergraduate Research Fellowship (SURF) program administered through the Office of Student-Faculty Programs at the California Institute of Technology. The authors thank Andre Stechert and Saleem Mukhtar for contributions to early prototypes of the Diamond Eye system.

References

[1] J.C. Aubele and E. N. Slyuta. "Small Domes on Venus: Characteristics and Origins". *Earth, Moon and Planets*, 50/51:493–532, 1990.

[2] M.C. Burl. "Continuously-scalable Template Models". (In review.), 1999.

[3] M.C. Burl, L. Asker, P. Smyth, U. Fayyad, P. Perona, L. Crumpler, and J. Aubele. "Learning to Recognize Volcanoes on Venus". *Machine Learning*, 30:165–194, 1998.

[4] M.C. Burl, C. Fowlkes, J. Roden, A. Stechert, and S. Mukhtar. Diamond Eye: A distributed architecture for image data mining. In *SPIE Thirteenth Intl. Symp. on Aerospace/Defence Sensing, Simulation, and Controls*, 1999. (To appear.).

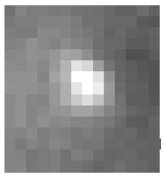
[5] M.C. Burl, T.K. Leung, and P. Perona. "Face Localization via Shape Statistics". In *Intl. Workshop on Automatic Face and Gesture Recognition*, 1995.

[6] M.C. Burl, T.K. Leung, and P. Perona. "Recognition of Planar Object Classes". In *Proceedings*

(a)

Model-type: Matched_filter
Name: volcano-foa.mdl
Target: volcano
Author: MCB
Date:
Version: 1.0
Comments: Average volcano derived from portion of Magellan Venus data. Illumination from left.

Filter:

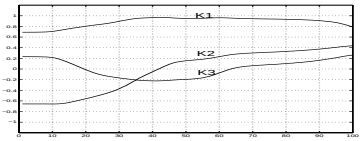


Threshold: 0.35

(b)

Model-type: CST
Name: scalable_crater.mdl
Target: bowl-floored crater
Author: MCB
Date:
Version: 1.0
Comments: Basis functions for continuously scalable bowl-floored crater. Prototemplate from Mars Viking imagery.
Nbasis: 3
Basis:

Scale_range: [1,2]
Scale_samples: 100
Interp:

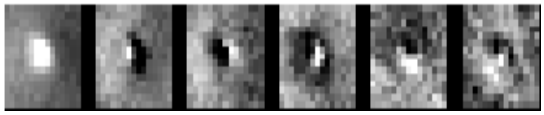


Threshold: 0.6

(c)

Model-type: PCA
Name: volcano-pca6.mdl
Target: volcano
Author: MCB
Date:
Version: 1.0
Comments: Basis functions for volcanoes from HOM4 subset of Magellan Venus dataset..

Nbasis: 6
Basis:



Classifier: @guassian_volcano-pca6.mdl

(d)

Model-type: SDCP
Name: frontal_face.mdl
Target: face
Author: MCB
Date:
Version: 1.0
Comments: Model for localization of frontal human faces. Model derived from caucasian male.

Nparts: 5
Parts: left-eye, right-eye, nose, left-mouth, right-mouth.
Detectors: @le.mdl, @re.mdl, @nose.mdl, @lm.mdl, @rm.mdl

Shape: @frontal_face_shape.mdl

Figure 2: Recognizers and query models can be accumulated and stored in the system database along with semantic information that will enable users to retrieve models by name. (a) Matched filter for Venusian volcanoes, (b) Continuously-scalable template model for a particular subclass of impact craters, (c) PCA model for Venusian volcanoes, (d) SDCP model for human faces.

of the IEEE Comp. Society Conf. on Computer Vision and Pattern Recognition, 1996.

[7] M.C. Burl and P. Perona. "Using Hierarchical Shape Models to Spot Keywords in Cursive Handwriting Data". In *Proceedings of the IEEE Comp. Society Conf. on Computer Vision and Pattern Recognition*, 1998.

[8] M.C. Burl, M. Weber, and P. Perona. "A Probabilistic Approach to Object Recognition Using Local Photometry and Global Geometry". In *Proceedings of the European Conf. on Computer Vision*, 1998.

[9] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* Wiley, 1973.

[10] M. Flickner, H. Sawhney, W. Niblack, et al. "Query by Image and Video Content – The QBIC System". *Computer*, 28(9):23–32, 1995.

[11] D. Forsyth, J. Malik, and R. Wilensky. "Searching for Digital Picutres". *Scientific American*, pages 88–93, June 1997.

[12] W. Freeman and E Adelson. "The Design and Use of Steerable Filters". *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13:891–906, 1991.

[13] H. Hotelling. "Analysis of a Complex of Statistical Variables into Principal Components". *J. Educ. Psych*, 24:417–441,498–520, 1933.

[14] A.E. Johnson and L.H. Matthies. Visual methods for small body exploration. URL. http://robotics.jpl.nasa.gov/people/johnson/.

[15] T.K. Leung, M.C. Burl, and P. Perona. "Finding Faces in Cluttered Scenes". In *Intl. Conf. on Computer Vision*, Cambridge, MA, 1995.

[16] T.K. Leung, M.C. Burl, and P. Perona. "Probabilistic Affine Invariants for Recognition". In *Proceedings of the IEEE Comp. Society Conf. on Computer Vision and Pattern Recognition*, 1998.

[17] J.L. Mundy and A. Zisserman. *Geometric Invariance in Computer Vision.* Artificial Intelligence. The MIT Press, 1992.

[18] H. Murase and S. Nayar. "Visual Learning and Recognition of 3-D Objects from Appearance". *Intl J. of Computer Vision*, 14:5–24, 1995.

[19] S. Nayar, S. Baker, and H. Murase. Parametric feature detection. In *Proceedings of the IEEE Comp. Society Conf. on Computer Vision and Pattern Recognition*, pages 471–477, 1996.

[20] P. Perona. "Deformable Kernels for Early Vision". *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(5):488–499, 1995.

[21] G.H. Pettengill, P.G. Ford, W.T.K. Johnson, R.K. Raney, and L.A. Soderblom. "Magellan: Radar Performance and Data Products". *Science*, 252:260–265, 1991.

[22] R.W. Picard and A.P. Pentland. "Introduction to the Special Section on Digital Libraries: Representation and Retrieval". *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(8):769–853, August 1996.

[23] I. Rigoutsos and R. Hummel. "A Bayesian Approach to Model-Matching with Geometric Hashing". *Computer Vision and Image Understanding*, 62(1):11–26, 1995.

[24] J. Roden and M.C. Burl. "Database Approaches for Image Mining Systems". (In review.), 1999.

[25] R.S. Saunders et al. "Magellan Mission Summary". *Journal of Geophysical Research*, 97(E8):13067–13090, August 1992.

[26] M. Turk and A. Pentland. "Eigenfaces for Recognition". *Journal of Cognitive Neuroscience*, 3(1), 1991.

[27] Virage. Visual Image Retrieval Engine. URL. http://206.169.1.82/market/vir.html.

[28] H.J. Wolfson. "Model-Based Object Recognition by Geometric Hashing". In *Proceedings of the European Conf. on Computer Vision*, pages 526–536, 1990.