Automating Image Processing for Scientific Data Analysis of a Large Image Database: Extended Report

Steve A. Chien and Helen B. Mortensen

S. Chien and H. Mortensen are affiliated with the Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Drive, Pasadena, CA, 91109-8099. Email: {steve.a.chien, helen.b.mortensen}@jpl.nasa.gov.

This work was performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. The MVP task was funded in part by the Operations/Artificial Intelligence Program, Code X, managed by Melvin Montemerlo and in part by the Continuous Improvement Initiative, Telecommunications and Mission Operations Directorate of JPL managed by Barbara Anderson.

Other past and present members of the MVP team are Christine Ying, Shouyi Hsiao, Forest Fisher, Darren Mutz, Alex Gray, Joe Nieten, and Jean Lorre.

This is an extended version of an article appearing in IEEE Transactions on Pattern Analysis, Vol. 18 No. 8, pp. 854-859.

Abstract

This article describes the Multimission VICAR Planner (MVP) system: an AI planning system which constructs executable image processing scripts to support Operational Science Analysis (OSA) requests made to the Jet Propulsion Laboratory (JPL) Multimission Image Processing Subsystem (MIPS). MVP accepts as input: 1. a set of image files; and 2. a high-level specification in science terms. MVP then produces an executable image processing script to fill the request. In producing this script, MVP must determine: 1. unspecified but required processing steps, 2. relevant image processing library programs, and 3. appropriate parameter settings for such programs. The MVP system embodies a general approach to representing and using knowledge of procedural tasks such as image processing. This article focuses on the general approach and the application of the approach to a specific area of image processing for planetary science applications involving radiometric correction, color triplet reconstruction, and mosaicking using the VICAR programming language. For this specific problem the MVP system significantly reduces the amount of effort required by image processing experts to fill a typical request.

Keywords

Data Analysis, Image Processing, Planning, Automated Programming

I. Problem Description

In recent times, improvements in spacecraft imaging hardware have caused a massive increase in the amount of scientific data and variety of science data products. Simultaneously, increased sophistication of instrumentation and image processing algorithms has greatly increased the knowledge required to prepare image data for analysis. While improvements in data storage and database technology have allowed physical access to the vast amounts of space-related data, preparing and processing available scientific data has become increasingly labor and knowledge intensive.

Development of general purpose data processing languages and interfaces can reduce this data access, preparation, and analysis problem. These languages and interfaces allow users to access and process data within a common environment. For image processing, the VICAR environment (Video Image Communication and Retrieval [1]) [16] is a major constituent of JPL's image processing capability. VICAR provides a standard interface

[1]This name is somewhat misleading as VICAR is used to process considerable non-video image data such as MAGELLAN synthetic aperture radar (SAR) data.

to allow a user to retrieve data and apply sophisticated data processing algorithms. The principal focus of the VICAR system is planetary imaging. Thus, VICAR supports imaging for JPL flight projects including VOYAGER, VIKING, MAGELLAN, GALILEO, and CASSINI. VICAR possesses many unique data processing capabilities relating to these image sources. VICAR has also been applied to other space imaging missions such as SIR-C and LANDSAT and instruments such as the Plasma Wave Spectrometer and other multispectral SAR datasets. The VICAR system has also been applied to numerous other applications including astronomy, earth resources, land use, biomedicine, and forensics. VICAR is a principal component of the Multimission Image Processing Laboratory (MIPL). Outside of JPL, VICAR users include universities, the military, research institutions, aerospace corporations, companies, and Home Institution Image Processing Subsystem (HIIPS) sites with over 100 users.

VICAR allows individual processing steps (called VICAR programs) to be combined into more complex image processing scripts called procedure definition files (PDFs). As one of their primary duties, JPL analysts construct PDFs to achieve tasks such as image correction, image enhancement, construction of mosaics, creation of movies and rendering of objects. Individual processing programs perform many different data processing functions such as described below:

1. *photometric correction* - the PHOTFUNC program can correct an image for lighting conditions due to the relative position of the lighting source, imaging device, and target;
2. *radiometric correction* - the FICOR77 program can correct VOYAGER images for varying camera response depending on camera state and other properties such as where in the field of view the image is read; and
3. *line fillin* - the GLLFILLIN program can interpolate missing lines in Galileo data caused by data transmission errors.

In order to fulfill OSA requests for image processing, analysts use their knowledge of the processing steps and processing program requirements to create VICAR scripts determining: the relevant programs to use, order of execution, and parameter settings.

Unfortunately, manual construction of VICAR scripts is both labor and knowledge intensive. Because of the complexity and amount of program knowledge relevant to the task as well as the many interacting problem goals, VICAR procedure generation is a labor intensive task. Generation of a highly complex VICAR procedure may take up to months of analyst time. The VICAR procedure generation problem is also a knowledge intensive task in that an analyst must possess knowledge of:

1. image processing and image processing programs (as of 1/93 there were approximately 50 frequently used programs, some having tens options)
2. database organization and database label information to understand the state of relevant data
3. the VICAR programming language to produce and store relevant information.

One difficulty facing analysts is the diversity of knowledge required to produce expert VICAR procedures. While certain VICAR users, such as expert analysts, may possess much of this knowledge, the vast majority of VICAR users are novice to one or more aspects of this knowledge. For example, a university user may know a great deal about the science behind the imaging and the theory behind the processing steps but may know little about the underlying assumptions of the implementation of the processing steps or VICAR itself. Similarly, a programmer who writes processing programs may be very knowledgeable about their particular program but may experience difficulty in writing a VICAR procedure to generate data to test their program. Unfortunately, this situation increases the load on experts who must spend a significant amount of their time assisting those less knowledgable. This great need for VICAR knowledge exists because of the significant time it takes to become proficient in multiple aspects of VICAR. Generally, a VICAR user with 1-2 years of experience is considered a novice VICAR user, while it may take 4-5 years to become a VICAR expert.

II. An Example of MVP Usage

In order to illustrate how MVP can be used to assist in VICAR planetary image processing, we now provide an example of MVP usage to ground the problem and the inputs and outputs required by MVP. Consider the three images, shown at the left of Figure 1. These images are of the planet Earth (northeastern Africa and Saudi Arabia) taken during the Galileo Earth 2 flyby in December 1992. However, many corrections and processing

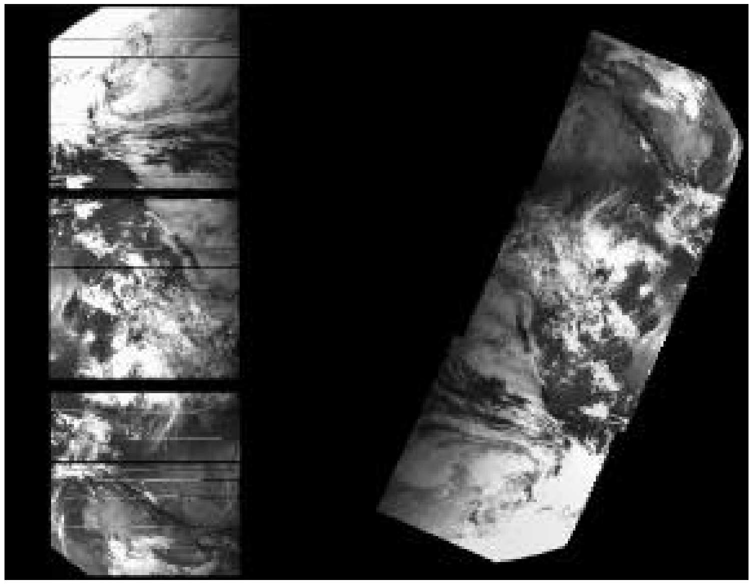
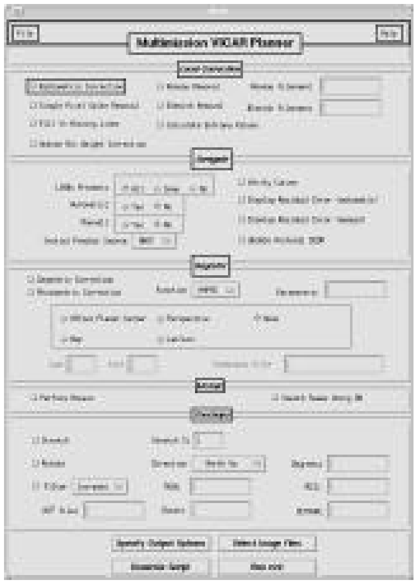


Fig. 1. Raw and Processed Image Files

steps must be applied before the images can be used. First, errors in the compression and transmission of the data from the Galileo spacecraft to receivers on Earth has resulted in a fair number of missing or noisy lines in the images. Line fillin and spike removals would correct most of these anomalies. Second, it is desirable to map project the images, in order to correct for the spatial distortion that occurs when a spherical body is represented on a flat surface. Third, in order to combine the images, we need to compute common points between the images and overlay them appropriately. Fourth, because we are combining multiple images which are likely to have been taken with different camera states, the images should be radiometrically corrected before combination. These are indicative of typical image processing goals which MVP addresses.

MVP enables the user to input image processing goals such as map projection and line fillin through the graphical user interface. Most of the image processing options we are interested in translate to toggle buttons on the interface. A few options require entering some text, usually function parameters that will be included as literals in the appropriate place in the generated VICAR script. Figure 2 shows the inputs to the MVP system. At left the graphical user interface is shown and at right a textual listing of the corresponding image processing goals passed from the interface to the MVP system.

After the image processing goals have been specified, MVP is ready to automatically



radiometric correction
pixel spike correction
fill in missing lines
uneven bit weight correction
no limbs present in images
perform automatic navigation
display automatic navigation residual error
perform manual navigation
display manual navigation residual error
map project with parameters...
mosaic images
smooth mosaic seams using dynamic range matching

Fig. 2. MVP Interface and Sample Problem Goals

generate the VICAR script. Using its knowledge of image processing procedures, the MVP planner constructs a plan of image processing steps which achieves the requested goal. After this process has terminated, this plan is translated into a VICAR script. This script, when run, performs the desired image corrections and constructs a mosaicked image of the three input files. The finished result of the image processing task is shown in Figure 1 on the right. The three original images now appear as a single mosaicked image, map projected with missing and corrupted lines filled in.

To further continue this example, shown in Figure 3 is a code fragment to perform portions of image navigation [2] for a Galileo image [3]. The higher-level conceptual steps (i.e., plan steps) are shown at the left and the corresponding VICAR code is shown at the right. In this case, the tasks being accomplished are acquiring initial navigation information, constructing initial overlap pairs, refining initial overlap pairs, checking for a previous tiepoint file, manually generating or refining tiepoints, and constructing the OM matrix for image navigation. In this case the overall user goal is to navigate the image. The other subgoals (and steps) are necessary to support this goal due to the dependencies

[2]Image navigation is the process of determining the matrix transformation to map from the 2-dimensional (line, sample) coordinate space of an image to a 3-dimensional coordinate space using information on the relative position of the imaging device (spacecraft position) and a model of the target being imaged (e.g., the planetary body).

[3]This code was generated by MVP.

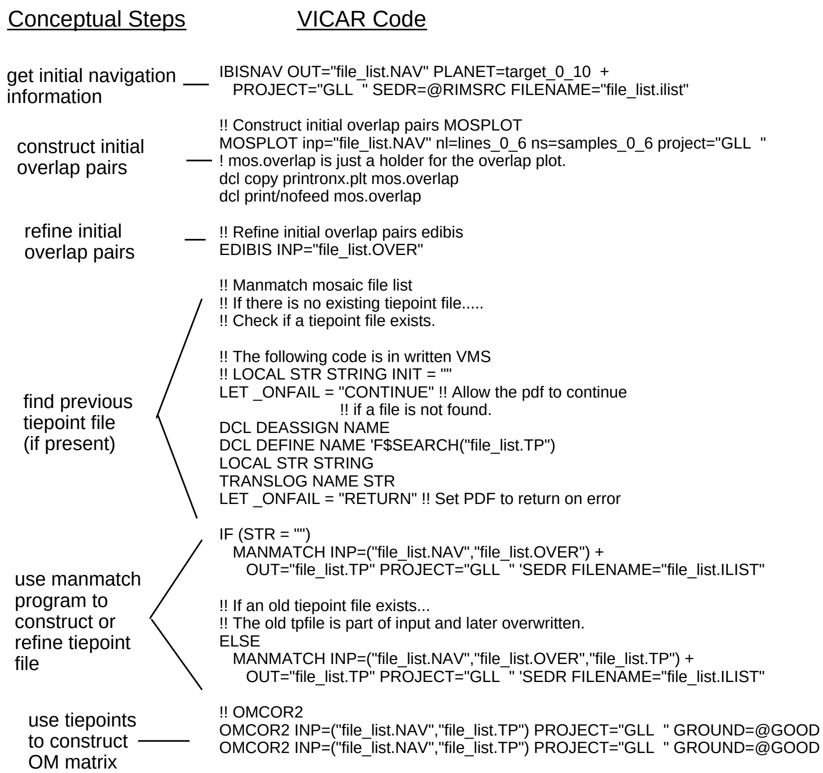


Fig. 3. Sample VICAR Code Fragment

of VICAR and image navigation.

Thus MVP allows the user to go directly from high level image processing goals to an executable image processing program. By insulating the user from many of the details of image processing, productivity is enhanced. The user can consider more directly the processing goals relevant to the end science analysis of the image, rather than being bogged down in the details such as file format, normalizing images, etc. The remainder of this article is organized as follows. First we describe the basic architecture of the MVP system. Next we describe the novel features of the MVP system from the perspective of planning technology. This is followed by a description of the significance and impact of the specific VICAR application as well as a discussion of the generality of the approach. Finally, we describe related work and conclusions.

III. The Multimission VICAR Planner (MVP)

MVP [4], [5] partially automates generation of image processing procedures from user requests and a knowledge-based model of an image processing area using Artificial Intelligence (AI) automated planning techniques [12], [19], [22]. In AI planning, a system uses: 1) a model of actions in a domain; 2) a model of the current state; and 3) a goal specification; to determine actions to achieve specified goals. In VICAR image processing, the actions are VICAR image processing programs, the current state is the current state of the image files of interest, and the specification of the desired state corresponds to the user image processing goals. By partially automating the filling of basic science image processing requests, image processing request turnaround time is reduced, analyst time is freed for more complex and challenging science requests, and analyst workload is reduced. As an additional benefit, encoding image processing knowledge in MVP allows valuable image processing knowledge to be retained by institutions, rather than being lost when analysts leave or retire.

From a technology standpoint, MVP is significant in several respects. First, MVP integrates multiple planning paradigms to most naturally represent domain constraints and human experts' problem-solving methods. Second, MVP uses novel methods to represent and reason about VICAR program option constraints. These technical contributions are described in the following sections. Third, MVP embodies an Artificial Intelligence planning approach to solving complex procedural automation problems such as automated image processing. We believe our approach is general and extends to other procedure automation tasks. In the discussion section we describe applications of the MVP engine to other procedure automation tasks to support this claim.

From an applications standpoint, MVP is significant in that MVP is a successfully deployed Artificial Intelligence Planning application which has had considerable impact in a specific VICAR planetary imaging task. Later sections describe the development and deployment of MVP as well as the impact of the MVP system.

A. The MVP Architecture

The overall architecture for the MVP system is shown in Figure 4. MVP uses two planning paradigms: decompositional planning [8], [14][4] and operator-based planning [2], [19]. Because these planning approaches are well understood in the planning literature, we focus on the adaptation of these methods in MVP to the image processing domain. For

[4]This approach has also been called Hierarchical Task Network Planning and Task Reduction planning.

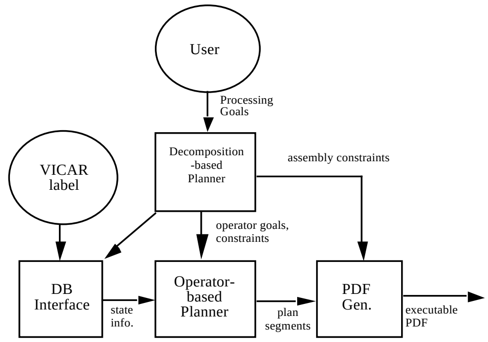


Fig. 4. MVP System Architecture

further information on these techniques the reader is referred to the specified references.

To use MVP, the user inputs a problem specification consisting of processing goals and image information using a menu-based graphical user interface. These goals and image information are then passed to the decomposition-based planner. The decomposition planner uses *decomposition rules* to implement two conceptual types of planning. First, the decomposition-based planner uses image processing knowledge to classify the overall problem type which the user has specified in a process called *skeletal planning* [12]. Second, the decomposition planner uses this classification to decompose the problem into smaller subproblems in a process called *hierarchical planning* [22].

The subproblems produced by the decomposition planner are then solved by *operator-based planning* [19], in which a planner uses a description of possible actions (in this case image processing steps) to solve subproblem goals as indicated by the problem decomposition. The resulting plan segments are then assembled using constraints derived in the decomposition process. The resulting plan is then used to generate an actual executable VICAR PDF using conventional macro expansion techniques.

From an AI planning technology standpoint, MVP uses both decomposition and operator-based planning techniques. MVP uses both techniques for two reasons: *search control* and *user understandability*.

The decomposition approach is needed for search control. Plans in the MVP domain can be of considerable length (up to 100 steps) and each step (or VICAR program) can involve reasoning about numerous complex effects (often operators have tens of effects)[5].

[5]However, it is worth noting that a VICAR script for a specific request generally does not contain complex control

Due to the large search space caused by this complexity, conventional operator-based planning approaches are not able to tractably construct plans in the VICAR domain without significant control knowledge. In the decomposition planning paradigm, it is natural to encode knowledge on how to break up a large problem into smaller subproblems. In the decomposition component, MVP breaks up a large search space planning problem caused by the complexity of the image processing problems into several smaller problems, thus reducing the search encountered during operator-based planning. Indeed, the problem decomposition rules used in MVP can be considered a very important form of search control knowledge essential to MVP's image processing capability.

MVP also uses decomposition-based planning for reasons of user understandability. Even if a purely operator-based planning approach were able to generate plans to solve the VICAR problems, these plans would be difficult for MIPL analysts to understand because MIPL analysts do not consider an entire image processing problem all at once. Typically, analysts begin by classifying the general problem being addressed into one of a general class of problems, such as mosaicking, color triple processing, etc. They then use this classification and the problem context to decompose the plan into several abstract steps, such as local correction, navigation, registration, touch-ups, etc. Because MVP uses decomposition-based planning to reduce the original image processing problem, MVP is able to produce an annotated trace of how the overall problem was classified and decomposed. This annotated trace greatly assists the analyst user in understanding the image processing plans generated by MVP.

B. Skeletal and Hierarchical Planning Using Decompositions

MVP integrates decomposition and operator-based planning paradigms. MVP first uses the decomposition planning framework to break an image processing problem into smaller subproblems – then solves the resulting subproblems using operator-based planning techniques. In order to break a problem into subproblems, MVP uses knowledge represented

constructs (e.g., conditionals, looping). Most VICAR scripts to fill a single request would contain few (if any) conditionals and these few conditionals are easily handled in the macro expansion phase. Most VICAR scripts for a specific request also do not contain loops. In the case where loops occur, they are generally looping over a finite set (such as over a known set of image files). Thus the more general, difficult problems of automated programming such as determining loop invariants and termination criteria are not relevant to the VICAR application domain.

as decomposition rules to perform two types of planning: *skeletal planning* and *hierarchical planning*.

In the following sections, we first describe the concepts of skeletal planning and hierarchical planning. We then describe how these concepts are implemented as decomposition rules in the decomposition planning paradigm.

B.1 Skeletal and Hierarchical Planning in MVP

Skeletal planning [12] is an approach to planning which casts planning as a structured classification problem. In skeletal planning, a planner identifies a new problem as one of a general class of problems based upon the goals and initial state. This technique was originally developed as a model of experiment design in molecular biology; however, skeletal planning is also an accurate model of how expert analysts attack VICAR procedure generation problems. Typically in a VICAR problem, there is a central goal for processing which then dictates a decomposition of the overall problem into subproblems. For example, a mosaicking problem decomposes into the subproblems of local correction, navigation, and registration. MVP attacks a VICAR problem by first determining the general problem class, and then using this problem class to perform an initial decomposition of the top-level image processing goals.

Hierarchical planning [22] is an approach to planning where abstract goals or procedures are incrementally refined into more and more specific goals or procedures as dictated by goal or procedure decompositions. MVP uses this approach of hierarchical decomposition to refine the initial skeletal plan into a specialized plan based on the specific current goals and situation. This allows the overall problem decomposition to be influenced by factors such as the presence or absence of certain image calibration files or the type of instrument and spacecraft used to record the image. For example, one common image processing step is geometric correction, in which a model of the target object is used to correct for different portions of the target being different distances from the imaging device. For VOYAGER images, geometric correction is performed as part of the local correction process, as geometric distortion is significant enough to require immediate correction before other image processing steps can be performed. However, for GALILEO images, geometric correction is postponed until the registration step, where it can be performed more efficiently.

LHS		RHS
G_I = initial goal set/actions		G_R = reduced goal set/actions
C_0 = constraints	==>	C_1 = constraints
C_2 = context		N = notes on decomposition

Fig. 5. Decomposition Rule Syntax

B.2 Implementing Skeletal and Hierarchical Planning in MVP using Decomposition Rules

MVP uses the decomposition planning paradigm [14], [8] to implement Skeletal and Hierarchical planning. In a decomposition approach, decomposition rules dictate how to attack and resolve flaws in a plan [6]. For example, a decomposition rule might specify a general method for attacking a particular goal, or how to break a problem into smaller problems. In many cases, it is possible to decompose a problem in several ways. In these cases, the planner then searches the space of possible decompositions. Decomposition approaches are extremely powerful in that many other paradigms (such as modal truth criterion planning [14]) can be implemented in a decomposition-based approach. The syntax for a decomposition rule is shown in Figure 5.

This rule states that a set of goals or actions G_I can be reduced to a new set of goals or actions G_R if the set of constraints C_0 is satisfied in the current plan and the context C_2 is satisfied in the current plan provided the additional constraints C_1 are added to the plan. Skeletal planning in MVP is implemented by encoding decomposition rules which allow for classification and initial decomposition of a set of goals corresponding to a VICAR problem class. The LHS of a skeletal decomposition rule in MVP corresponds to a set of conditions specifying a problem class, and the RHS specifies an initial problem decomposition for that problem class. For example, Figure 6 shows a decomposition rule for the problem class *mosaicking with absolute navigation.*

The simplified decomposition rule shown in Figure 6 states that if mosaicking is a goal of the problem and an initial problem decomposition has not yet been made, then the initial problem decomposition should be into the subproblems local correction, navigation,

[6]For example, an unachieved goal could be considered a plan flaw and decomposition rules might specify ways to achieve the goal. Alternatively, a negative interaction between two steps might be a flaw – decomposition rules might specify ways in which the interaction could be resolved.

LHS		RHS
G_I= mosaicking goal present	G_R =	1. local correction, 2. navigation 3. registration 4. mosaicking 5. touch-ups
C_0= null	C_1 =	these subtasks be performed in order 1. 2. 3. 4. 5. protect local correction until mosaicking
C_2= an initial classification has not yet been made	N =	problem class is mosaicking

Fig. 6. Skeletal Planning Decomposition Rule

etc. and that these steps must be performed in a certain order. This decomposition also specifies that the local correction goals must be protected during the navigation and registration processes. In general, MVP permits goals and abstract steps to be specified in the G_I and G_R fields. The left hand side constraints C_0, right-hand side constraints C_1, and context C_2 specify restrictions on when the rule is applicable, and include :

1. constraints on the ordering of steps or goals;
2. constraints on the assignment of variables representing objects in the plan;
3. goals or steps that must be present in the plan; and
4. goals or steps that must not be present in the plan.

These constraints and context specify restrictions on the situations in which G_I can be performed (or solved) by performing (solving) G_R. As such these restrictions are a convenient place to encode search control information.

Hierarchical planning is also implemented within the decomposition framework. In this case the LHS specifies a context in which a set of goals or actions can be decomposed into a lower level set of goals or actions. For example, the decomposition rule in Figure 7 states that if the limb is present in all of the images (meaning that the sun-facing edge of the planet is visible in all of the images), for VOYAGER and GALILEO images, the navigation step can be performed by absolute navigation (a process in which each of the images can be navigated independently).

This decomposition-based approach to skeletal and hierarchical planning in MVP has several strengths. First, the decomposition rules very naturally represent the manner in which the analysts attack the procedure generation problem. Thus, it was a relatively straightforward process to get the analysts to articulate and accept classification and

LHS	RHS
G_I= navigation action present	G_R = 1. absolute navigation
C_0= null	
C_2= the project is VOYAGER or GALILEO and limb present in all images	C_1 = null
	N = null

Fig. 7. Hierarchical Refinement Decomposition Rule

decomposition rules for the subareas which we have implemented thus far. Second, the notes from the decomposition rules used to decompose the problem can be used to annotate the resulting PDF to make the VICAR programs more understandable to the analysts. Third, relatively few problem decomposition rules are easily able to cover a wide range of problems and decompose them into much smaller subproblems.

C. Operator-based Planning in MVP

MVP uses classical operator-based planning techniques to solve subproblems produced by the decomposition-based planner. An operator-based planner uses:

1. a model of actions M (in this case the model represents the requirements and effects of individual VICAR steps);
2. a specification of a current state C (this corresponds to the current database state); and
3. a specification of a goal criteria G (this corresponds to user request specification)

to derive:

a sequence of actions A, that when executed in the current state C, result in a state which satisfies the goal criteria G. In this case A will correspond to the VICAR script the user can execute to perform the image processing task at hand.

In operator-based planning, an action is represented in terms of its preconditions (required to be true before an action can be executed), and its effects (true after an action is executed). For example, in VICAR image processing, the program GALSOS is used to radiometrically correct Galileo image files. This would be represented by a planning action for the GALSOS program, which could be applied to an image file. This action would have the precondition that the image file be a Galileo image file. This action would also have the effect that the image file is radiometrically corrected after GALSOS has been run. The GALSOS operator definition is shown below.

```
operator GALSOS
        :parameters ?infile ?ubwc ?calc
        :preconditions
                the project of ?infile must be galileo
```

```
                the data in ?infile must be raw data values
    :effects
                reseaus are not intact for ?infile
                the data in ?infile is not raw data values
                missing lines are not filled in for ?infile
                ?infile is radiometrically corrected
                the image format for ?infile is halfword
                ?infile has blemishes-removed
                if (UBWC option is selected) then ?infile is uneven bit weight corrected
                if (CALC option is selected) then ?infile has entropy values calculated
```

When constructing a plan to achieve a goal G1, a planner will consider those actions which have G1 as an effect. Thus, if the planner wanted to achieve that a particular image file was radiometrically corrected, it would consider applying the VICAR program GALSOS on the image file. If a planner decides to add an action A1 to a plan to achieve a goal, it will then have to achieve all of the preconditions of A1 in a process called *subgoaling*. For example, the VICAR program PTP requires that an image file be in byte format before PTP can be applied. Thus if the planner decides that it wants to apply the PTP program to a file, it then must ensure that the image file is in byte format. In some cases this will already be true, in other cases running a program to change the file format may be required.

Planning is also complicated by the fact that there are typically interactions between subparts of the plan. Thus, actions introduced to achieve goals in one part of the plan may undo goals achieved in another part of the plan. The process of ensuring that such interactions do not occur is called *protection*. Protection can involve such measures as ensuring that the goal is no longer needed when it is undone, or ensuring that the offending action effect does not in fact refer to the same object as the achieved goal (by creating a copy of a file, for example). Because operator-based planning is not unique to MVP, we have only briefly sketched the key elements of operator-based planning, for a more detailed treatment of operator-based planning algorithms the reader is referred to [19], [2].

C.1 Representing and Reasoning about Program Options in VICAR

One novel aspect of the VICAR domain is that a significant portion of the search to achieve goals and to enforce protections is not at the *program selection level* (which corresponds to operator selection in the planning process) but rather at the *program option selection level* (which corresponds to the operator effect planning level). Thus, when planning to achieve a goal, MVP searches more to determine how to set program options to achieve a goal (e.g. how to set variable constraints to satisfy preconditions) rather than to determine which VICAR program (planning operator) to use to achieve the goal. This presents a problem for efficiently reasoning about interacting program options (operator effects) in that certain combinations of program options (operator effects) are inconsistent (i.e., cannot be used together).

For example, when considering operators to achieve a perspective correction, MVP might need to consider a family of map projection programs (MAP, MAP3, and MAP4) and the rotational correction program PTP. This search to find the correct program is not overly difficult. However, for a particular problem, after having selected PTP, MVP would need to determine which method to use to specify the spacecraft pointing information. There are ways in which the pointing information could be computed. MVP could use existing navigation data, from one of many sources: FARENC, basic NAIF, AVIS, etc. Alternatively, MVP could regenerate navigation data from basic NAIF using one of many methods including NEARENC, FARENC, MANMATCH, AUTOMATCH. Each of these methods has its own range of applicability and interactions with other parts of the problem. Each of these methods is represented as a set of program options specifications and preconditions for a conditional effect of the operator. Thus, using the existing FARENC data would have certain preconditions and program option settings which might be incompatible with other program options. Thus an operator will have a set of condition-effect pairs (condition C effect E). The semantics of this conditional effect are that if the relevant conditions C are met when the operator is executed (the so-called conditions) the effect E will be true in the resulting situation.

In most cases, it is not immediately apparent which method is appropriate. In this situation MVP must perform search among the possible options. Consequently, the ability to search these combinations of operator effects efficiently when the operator effects do not interact while correctly restricting to those legal combinations is a unique capability of the MVP planner and represents a capability that other operator-based planners do not have.

Due to this difficulty of search among VICAR program options, MVP uses an operator-based planning component which extends conventional operator-based planning by representing VICAR program options as variable codesignation constraints. Thus, if a VICAR program has a program option which allows for several ways to specify spacecraft pointing information for a particular image processing step, MVP would represent these different methods as conditional effects of a single planning operator, with the appropriate preconditions (including variable codesignations). If certain program options (operator effects) are inconsistent, they would be represented by having preconditions with conflicting codesignation constraints. When using an operator effect to achieve a subgoal in the plan, MVP first checks to see if the codesignation preconditions are consistent with the plan, only then allowing the effect to be used (and adding the codesignation constraints to the plan).

For example, returning to our PTP example, the PTP program allows for multiple images taken at similar times to be corrected to appear as if they were taken at the same time. This program needs to know the position of the spacecraft relative to the target of the image (typically the planet center). This information can be specified in one of several different ways, such as using the spacecraft navigation information, specific VICAR programs which attempt to compute this information from the image (the usual method), or by specifying the exact pixel location known from previous operations in the PDF. Typically, an analyst will include VICAR code to derive this information directly from the image. In this case the exact program and options being used to compute this information are frequently needed by the PTP program. For example, for the VOYAGER project, if one wishes to use pointing information previously derived using the FARENC program, it would be stored in a navigation data structure called SEDR. A simplified representation of this conditional effect is shown below (codesignation constraints are marked by an asterisk *).

```
IF
   (the SEDRSRC pointing specification is used
   and SEDRSRC is specified to be FARENC)* and
   (the PC and RPC pointing specification is not used)* and
   the project of file is VOYAGER1 or VOYAGER2 and
   appropriate SEDR data files for file exist and
   the camera number RCAM for the file has been correctly specified and
   the FDS for file has been correctly specified
THEN
   the output image outfile will be registered to the
   reference image as specified
```

This method for representing VICAR program options is important in that it allows for independent program options to be reasoned about and constrained independently yet represents the interaction between conflicting options. For example, the PTP program option to translate the image during the PTP step, requires that the camera pointing specification be directly specified using the planet center (PC) and reference planet center options (RPC), which specify a particular point in the image directly as the planet center. These options are incompatible with the FARENC source of camera pointing information. MVP represents this constraint by negative codesignations appearing in the preconditions of these incompatible options (the *-ed codesignation constraints listed above). However, non-interacting options such as PTP options to resize the image or to include or delete the background of the image are not affected. These options do not interact with the specification of pointing information and thusly can be reasoned about independently.

In contrast, most planners do not allow for codesignation constraints on operator effects. Consequently, in order to represent incompatible program options they would need to use either: 1. contradictory preconditions to enforce disallowed combinations; or 2. break inconsistent operator effects into different planning operators (with each operator representing a consistent combination of operator effects). Option 1 would require the ability to easily detect inconsistent preconditions when choosing an effect and is analogous to our codesignation method (but more complicated). Not detecting these contradictory preconditions when choosing an effect would cause considerable unnecessary search. Option 2 (breaking inconsistent effects (program options) into separate operators) requires an increase in the number of operators exponential in the number of inconsistent options (N pairs of incompatible options requires 2^N operators). Even worse, when selecting an operator which one option decided, the planner would have to arbitrarily commit to decisions on other program options - potentially causing unnecessary search. As the number of program options can be quite large (frequently in the tens of options), these are important representational and search efficiency issues.

C.2 An Example of Subgoaling in VICAR Image Processing

To illustrate how the operator-based planning process performs subgoaling, consider the subgoal graph illustrated in Figure 8.[7] In this case the user has selected the goal that the

[7]The VICAR code previously shown in Figure 3 is taken from this example.

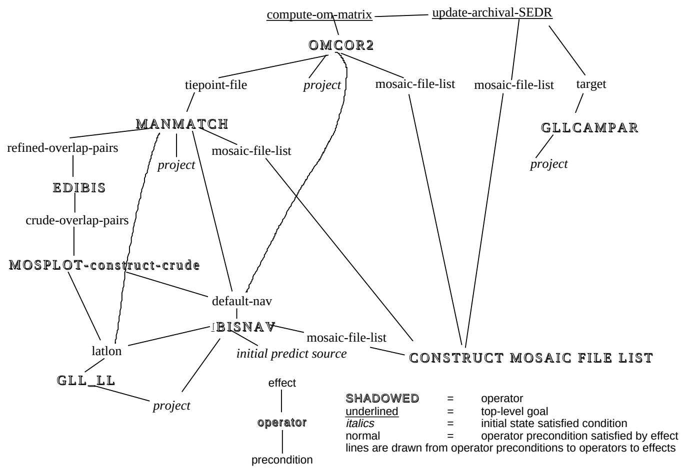


Fig. 8. Subgoal Graph for Manual Relative Navigation of Galileo Image Files

images be navigated using manual methods and that the archival navigation information for the image should be updated. The decomposition planner has access to the knowledge that in order to navigate the image, the operational goal is to construct an OM matrix which defines the transformation from (line, sample) in the image to some known frame of reference (usually the position relative to the target planet center). The planner knows that in order to compute this matrix it must have a tiepoint file, the project of the image, and the image files formatted into a mosaic file list. In order to produce a tiepoint file for the goal specification of manual navigation, the planner uses the MANMATCH program. The MANMATCH program in turn requires a refined overlap pairs file, the project of the images, the initial predict information, and again a mosaic file list. The refined overlap pairs file can be constructed using the EDIBIS program, but this requires a crude overlap pairs file based on an initial predict source. This crude overlap pairs file in turn requires the default navigation method, and the latitude and longitude of sample image files. The rest of the graph is generated similarly. This subgoal graph is generated in response to the particular combination of user goals and the state of the selected image files.

C.3 An Example of Resolution of Goal Conflicts in VICAR Image Processing

To illustrate how the operator-based planning process resolves interactions between steps, consider the (simplified) image processing operators shown in Figure 9. The relevant operators to achieve the goals of missing line fillin, spike removal, and radiometric

Operator	VGRFILLIN	GLLFILLIN	ADESPIKE	FICOR77	GALSOS
Preconditions	VGR image EDR present	GLL image	(GLL image) or ((VGR image) and (raw values))	VGR image	GLL image raw pixel values
Effects	missing lines filled in.....		spike removal not raw values	radiometric corr. blemish removal not raw values	radiometric corr. reed-solomon overflow corr. saturated pixel corr. not missing line fillin

Fig. 9. Simplified Operator Definitions

correction for Voyager and Galileo images are shown below. When constructing a plan to achieve these goals, depending on the project of the image file (e.g., either Voyager or Galileo), MVP determines the correct program to use because the preconditions enforce the correct program selection.

	VICAR Program		Execution Order	
Goal	Voyager	Galileo	Voyager	Galileo
fillin missing lines	VGRFILLIN	GLLFILLIN	VGRFILLIN	GALSOS
remove spikes	ADESPIKE	ADESPIKE	ADESPIKE	GLLFILLIN
radiometric corr.	FICOR77	GALSOS	FICOR77	ADESPIKE

However, determining the correct ordering of actions can sometimes be complex. In this case, the correct order to achieve the goals of line fillin, spike removal, and radiometric correction is dependent upon the project of the file. In the case of Voyager files, ADESPIKE (spike removal) requires raw pixel values and FICOR77 (radiometric) changes pixel values to correct for camera response function – thus FICOR77 removes a necessary condition for ADESPIKE (raw pixel values). This interaction can be avoided by enforcing that ADESPIKE occurs before FICOR77. Additionally, VGRFILLIN requires binary EDR header on the image file, and ADESPIKE removes the binary EDR header, thus ADESPIKE removes a necessary condition for VGRFILLIN. This interaction can be avoided by requiring VGRFILLIN to be executed before ADESPIKE. Thus in the VOYAGER example the only legal execution order is VGRFILLIN, ADESPIKE, FICOR77.

In the Galileo case, GALSOS undoes missing line fillin (the goal achieved by the GLLFILLIN operator). Thus in order to avoid undoing this processing, GLLFILLIN must be applied after GALSOS. Additionally, GALSOS requires raw pixel values, and ADESPIKE alters the pixel values, so ADESPIKE removes a necessary condition for GALSOS. This interaction can be avoided by requiring that GALSOS occurs before ADESPIKE.

This simple example illustrates some of the interactions and context-sensitivity of the

Fig. 10. Problem Space Information

Problem Space	# operators	goals	typ. search
local correction	15	7	60
automatic navigation	20	4	150
manual navigation	24	4	300
photometric correction	5	2	60
registration	13	5	110
mosaicking	4	3	325
touch ups	10	3	325

VICAR image processing application. All of these interactions and context sensitive requirements are derived and accounted for automatically by MVP using the operator specification, thereby allowing plan construction despite the presence of complex interactions and conditions.

D. On the Impact of Combining Decomposition and Operator-based Planning Methods

One obvious question is the impact of combining decomposition and operator-based planning methods. Earlier in the article, we stated that the two reasons for combining decomposition and operator-based methods methods were user understandability and search control. While it is difficult to quantify the effectiveness of increased understandability of plans, in this section we attempt to roughly quantify the effectiveness of decomposition methods in controlling the search required by the operator-based planner.

The principle impact of decomposition planning on search is to decompose the planning process into independent subproblems which can be solved independently in a known sequential fashion. In the current MVP knowledge base, there are seven such problem spaces: local correction, automatic navigation, manual navigation, photometric correction, registration, mosaicking, and touch-ups. In Figure 10, we describe the salient information on each of the problem-spaces. First, for each problem space we list the number of relevant planning operators and top-level input goals as this gives some indication of the size of the problem space. We also list the typical number of plans searched in the problem space.

The overall effect of decomposition planning on search is to break down the search into more manageable subproblems. For example, if subproblem A typically requires searching α plans and subproblem B typically requires searching β plan, solving both problems simultaneously might require on order $\alpha\beta$ plans[8]. Overall, because the search spaces combine (roughly!) multiplicatively, the impact of adding domain knowledge to decompose subproblems has been enormous. For example, originally the automatic navigation and manual navigation problem spaces were represented as a single navigation problem space. However, this problem space required too much search (on the order of 50,000 plans), so it was broken into the automatic and manual navigation problem spaces.

IV. Significance of this Work

In this section we describe the significance of the MVP system from two perspectives. First, MVP represents a successful solution to a specific problem area in planetary image processing. Second, the planning technology used in MVP represents a general methodology for automating procedure generation tasks. We describe some evidence from multiple applications that this planning technology is applicable to other domains.

MVP is significant in that it is a successful deployed solution to a specific constrained planetary image processing problem. MVP2.0 is implemented in C and runs on Sun SparcStations under Unix and Motif and under VMS on Vaxes. MVP is currently operational and available for use by analysts at JPL's Multimission Image Processing Laboratory (MIPL) for the general areas of radiometric correction, color triplet reconstruction, mosaicking with relative or absolute navigation, registration, and simple filtering and stretching tasks. MVP supports roughly 70 VICAR subroutines. The MVP knowledge base includes about 50 operators, 50 decomposition rules, and tracks roughly 70 attributes per image file. The produced plan for a complete problem may contain over 100 operators, with a typical plan containing perhaps 60 planning operators. These 60 planning operators would generally correspond to a 100 line VICAR script.

We now specifically describe the types of image processing problems which MVP currently covers. In terms of local correction, MVP performs the following types of correction: Radiometric Correction - correction for the camera state when imaging, Spike Removal - smoothing of images, Missing Line Fillin - correcting for missing lines typically from

[8]This is clearly a simplification, it might require less search than this because weak heuristics might tend to guide the search well. However, it might be worse because adding the second subproblem might weaken heuristics that work well for the first subproblem alone. Empirically in the MVP image processing application combining two search spaces A and B as above would result in search slightly less than $\alpha\beta$.

transmission errors. Uneven Bit Weight Correction - correction using Galileo error correction codes. Reseau Removal - correcting for Voyager camera artifacts, Blemish Removal - correcting for Galileo and Voyager camera artifacts, and Entropy Value Calculation - measuring the effectiveness of Galileo coding schemes.

In the Automatic and Manual Navigation Phases, MVP supports the following image processing operations. As part of the basic computation of the required matrix transformation to recover 3-D imaging information using spacecraft and target body positioning information, MVP supports several capabilities: Navigation Images using Limb finding - using the illuminated portion of the planet to determine the relative position of the planet, Verification of Limbs Using the CURVES program, Automatic Navigation using VICAR Automatch Feature Finding Software, Manual Navigation using VICAR Manmatch Feature Finding Software, Calculation of initial Navigation information using numerous JPL methods, Display of Residual Error during Each Step of the Navigation Process, and Updating Archival SEDR.

In the Registration Phase, the image can be translated, rotated, and otherwise transformed in a variety of ways including: translation of planet center, perspective correction - correcting for planetary or atmospheric rotation in order to combine images taken at different times, translation based on coordinate references such as latitude and longitude, map projection, photometric correction and geometric correction.

MVP also supports mosaicking to combine multiple images into a single image using navigation and registration information from previous phases. Seam smoothing based on normalization of dynamic ranges is also supported.

In the touch-up phase, MVP supports stretching dynamic range based on various functions (e.g., linear, gaussian, percentile), rotation of images, and several forms of filtering - high pass, low pass, and multiple varieties of modulation transfer function filtering.

It is worth noting that MVP does not fully automate this planetary imaging task. In typical usage, the analyst receives a request, determines which goals are required to fill the request, and then runs MVP to generate a VICAR script. The analyst then runs this script and then visually inspects the produced image(s) to verify that the script has properly satisfied the request. In most cases, upon inspection, the analyst determines that some parameters need to be modified subjectively or goals reconsidered in context. This process typically continues several iterations until the analyst is satisfied with the image product.

In order to assess the impact of the MVP system, we asked analysts to estimate the effort required to satisfy typical requests involving radiometric correction, color triplet reconstruction, and mosaicking with relative or absolute navigation, registration, and simple filtering and stretching. The analysts estimated that for these tasks MVP reduces effort to generate an initial PDF for an expert analyst from 1/2 a day to 15 minutes and reduces the effort for a novice analyst from several days to 1 hour. This represents over an order of magnitude in speedup. The analysts also judged that the quality of the PDFs produced using MVP are comparable to the quality of completely manually derived PDFs. While these results certainly do not represent a rigorous controlled empirical study (which would be difficult due to the variability of requests and the scarcity of analyst time), these results do represent strong evidence as to the usefulness of the MVP system.

From another perspective, the MVP system represents a general approach of using Artificial Intelligence Planning Technology as a solution to procedure automation problems. On this front, the MVP planning technology has been adapted to a different domain, that of generating procedures to operate JPL Deep Space Network Antennas to communicate with spacecraft [10], [11], [7]. This system, called DPLAN, was demonstrated in February 1995 [11] and will be operational at multiple Deep Space Network complexes by the Fall of 1996. In a separate effort, work is underway to field another version of MVP for a planetary geology group at the department of Geology at Arizona State University. This version of MVP would be used primarily for map projection and detection and attribute measurement of geological features in Synthetic Aperture Radar (SAR) image data. In the MVP JPL application, DSN application, and MVP/ASU application, we were able to naturally encode planning knowledge in the form of decompositions and operators. As a result of this experience in representing several domains using the MVP representation constructs we feel that the techniques and representations have some generality.

V. Related Work and Conclusions

Related work can be broadly classified into: related image processing languages, related automated image processing work, and related AI planning work. In terms of related image processing languages, there are many commercial and academic image processing packages - such as IDL, Aoips,and Merlyn. Generally, these packages have only limited ability to automatically determine how to use different image processing programs or algorithms based on the problem context (e.g., other image processing goals and initial image state). These packages only support such context sensitivity for a few pre-anticipated cases.

However, there are several previous systems for automatic image processing that use a domain independent mechanism. Work at the Canadian Centre for Remote Sensing (CCRS) [3] has been towards a case-based system for image processing and acquisition of image processing knowledge. This work differs from MVP in that they use a case-based reasoning approach in which an existing image processing problem is solved by retrieving a previous problem and solution and adapting it to solve the current problem. Grimm and Bunke [9] developed an expert system to assist in image processing within the SPIDER library of image processing routines. This system uses many similar approaches in that: 1. it classifies problem types similar to the fashion in which MVP performs skeletal planning; and 2. it also decomposes larger problems into subproblems which MVP performs in decomposition planning. This system is implemented in a combination of an expert system shell called TWAICE (which includes both rules and frames) and Prolog. This very basic implementation language gives them considerable power and flexibility but means that their overall system uses a less declarative representation than our decomposition rules and operators which have a strict semantics [8], [2]. Previous work on automating the use of the SPIDER library includes [21] which performs constraint checking and step ordering for a set of conceptual image processing steps and generation of executable code. This work differs from MVP in that: 1. they do not infer missing steps from step requirements; 2. they do not map from a single abstract step to a context-dependent sequence of image processing operations; and 3. they do not reason about negative interactions between subproblems. MVP has the capability to represent and reason about all 3 of these cases. Other work by Jiang and Bunke [13] involves generation of image processing procedures for robotics. This system performs subgoaling to construct image processing plans. However their algorithm does not appear to have a general way of representing and dealing with negative interactions between different subparts of the plans. In contrast, the general Artificial Intelligence Planning techniques used by MVP use conflict resolution methods to guarantee correct handling of subproblem interactions.

Other work by Zmuda [23] describes work in automatically deriving classification software by using machine learning techniques. However for the MVP applications, the search space of possible programs is too large and there is no end feedback (as in classification) to drive the learning process. Another piece of related work is the SATI system [1] which uses an interactive dialogue with the user to drive an automated programming approach to generating code to satisfy the user request. OCAPI [6], a semantically integrated automated image processing system, while being very general provides no clear way to represent the large number of logical constraints associated with the problems MVP was designed to solve. Another image processing system [17] provides a means for representing knowledge of image analysis strategies in an expert system but does not use the more declarative AI planning representation. Perhaps the most similar planning and image processing system is COLLAGE [15]. The COLLAGE planning differs from MVP in that COLLAGE uses solely the decomposition approach to planning. COLLAGE differs from MVP in the applications sense in that it focuses primarily on earth imaging applications in the Khoros environment, where MVP has focused on planetary applications in the VICAR environment.

Other related work in automatic image processing focuses on speeding execution of algorithms [18], [20] through parallelism but requires that the image processing plans be manually constructed into task networks whereas MVP automatically constructs the task network from the goal specification and initial image state information.

From the standpoint of planning technology, MVP differs from other planning work in two ways. First, it integrates decomposition-based (also called hierarchical task network) and operator-based approaches to more closely model how human experts solve image processing problems. Second, it uses an explicit constraint model to efficiently search among operator effects (which correspond to VICAR program options).

This paper has described the application of AI planning techniques to automate image processing as embodied by the MVP system. This work is a significant advance in the state of the art in AI planning technology in that: 1. it represents an integration of decomposition and operator-based planning paradigms; and 2. it uses explicit constraints to efficiently reason about operator effects. The AI planning approach represents a general approach to automating procedure generation problems and we presented evidence to support this view. The work described in this paper is also significant from an image processing applications perspective. MVP2.0, a fielded planning system reduces the effort to perform the specific VICAR image processing tasks of: radiometric correction, color triplet reconstruction, and mosaicking for experts from 4 hours without MVP to 15 minutes with MVP. This successful application is being expanded to cover additional areas of image processing and fielding to remote university image processing sites.

References

[1] O. Capdevielle, P. Dalle, "Image Processing Chain Construction by Interactive Goal Specification," Proceedings of the First IEEE Int. Conf. on Image Processing, Austin, TX, Nov 1994, Vol. 3, pp. 816-819.

[2] D. Chapman, "Planning for Conjunctive Goals, 1987, "*Artificial Intelligence 32*, 3.

[3] D. Charlebois, J. DeGuise, G. Goodenough, S. Matwin, and M. Robson, "A Case-based Planner to Automate Reuse of ES Software for Analysis of Remote Sensing Data," Internation Geoscience and Remote Sensing Symposium (IGARSS), Vol 3, pp. 1851-1854, 1991.

[4] S. Chien, "Using AI Planning Techniques to Automatically Generate Image Processing Procedures: A Preliminary Report," Proc. 2nd Int. Conf. on AI Planning Systems, Chicago, IL, June 1994, pp. 219-224.

[5] S.Chien,"Automated Synthesis of Image Processing Procedures for a Large-scale Image Database," Proc. First IEEE Int. Conf. on Image Processing, Austin, TX, November 1994, Vol. 3, pp. 796-800.

[6] V. Clement and M. Thonnat, "A Knowledge-based Approach to Integration of Image Processing Procedures," Image Understanding, 57:166-184, March 1993.

[7] S. Chien, X. Wang, T. Estlin, and A. Govindjee, "Automatic Generation of Temporal Dependency Networks for Antenna Operations," submitted to Telecommunications and Data Acquisition.

[8] K. Erol, J. Hendler, and D. Nau, "UMCP: A Sound and Complete Procedure for Hierarchical Task Network Planning," Proceedings of the 2nd International Conference on AI Planning Systems, Chicago, IL, June 1994, pp. 249-254.

[9] F. Grimm and H. Bunke, "An Expert System for the Selection and Application of Image Processing Subroutines, " Expert Systems, May 1993, Vol. 10, No. 2, pp. 61-74.

[10] R. Hill, S. Chien, C. Smyth, and K. Fayyad, Planning for Deep Space Network Operations, Proceedings of the 1995 AAAI Spring Symposium on Integrated Planning Applications, Stanford, CA, March 1995, pp.51-56.

[11] R. Hill, S. Chien, K. Fayyad, C. Smyth, T. Santos, R. Chen and R. Bevan, "Sequence of Events Driven Automation of The Deep Space Network," Telecommunications and Data Acquisition, Winter 1995.

[12] Y. Iwasaki and P. Friedland, "The Concept and Implementation of Skeletal Plans,"Journal of Automated Reasoning 1, 1 (1985), pp. 161-208.

[13] X. Jian and H. Bunke, "Vision Planner for an Intelligence Multisensory Vision System," Technical Report, University of Bern (extended version of a paper appearing in ICPR 1994).

[14] A. Lansky, Localized Planning with Diverse Plan Construction Methods, Technical Report FIA-93-17, NASA Ames Research Center, June 1993.

[15] A. Lansky, M. Friedman, L. Getoor, S. Schmidler, and N. Short Jr., "The Collage/Khoros Link: Planning for Image Processing Tasks," Proceedings of the 1995 AAAI Spring Symposium on Integrated Planning Applications, March 1995, pp. 67-76.

[16] S. LaVoie, D. Alexander, C. Avis, H. Mortensen, C. Stanley, and L. Wainio, VICAR User's Guide, Version 2, JPL Internal Document D-4186, Jet Propulsion Lab., California Institute of Tech., Pasadena, CA, 1989.

[17] T. Matsuyama, "Expert Systems for Image Processing: Knowledge-Based Composition of Image Analysis Processes," Computer Vision, Graphics, and Image Processing 48, (1989), pp. 22-49.

[18] M. Moore, G. Karsai, and J. Sztipanovits, "Model-based Programming for Parallel Image Processing," Proceedings of the First IEEE International Conference on Image Processing, Austin, TX, Nov 1994, Vol. 3, pp. 811-815.

[19] J. S. Pemberthy and D. S. Weld, "UCPOP: A Sound Complete, Partial Order Planner for ADL," Proceedings of the 3rd International Conference on Knowledge Representation and Reasoning, Oct 1992.

[20] P. Romig, A. Samal, "Devious: A Distributed Environment for Vision Tasks," Proceedings of the First IEEE International Conference on Image Processing, Austin, TX, Nov 1994, Vol. 3, pp. 786-790.

[21] K. Sakaue and H. Tamura, "Automatic Generation of Image Processing Programs by Knowledge-based Verification," IEEE Conference on Computer Vision and Pattern Recognition, pp. 189-192, 1985.

[22] M. Stefik, "Planning with Constraints (MOLGEN: Part 1)," Artificial Intelligence 16,2(1981), pp. 111-140.

[23] M. Zmuda, M. Rizki, and L. Tamburino, "Approaches to Synthesizing Image Processing Programs," IEEE National Aerospace and Electronics Conference, Vol. 3, pp. 1054-1059, 1991.

Steve Chien is Technical Group Supervisor of the Artificial Intelligence Group, Advanced Information Systems Section at the Jet Propulsion Laboratory, California Institute of Technology where he leads efforts in automated planning and scheduling for: science data analysis, spacecraft operations, spacecraft design, Deep Space Network Antenna operations, and maintenance of space transportation systems. Dr. Chien is also an Adjunct Assistant Professor with the Department of Computer Science of the University of Southern California. He holds B.S., M.S., and Ph.D. degrees in Computer Science, all from the University of Illinois. Dr. Chien is a recipient of the 1995 Lew Allen Award for Excellence, JPL's highest award recognizing outstanding technical achievements by JPL personnel in the early years of their careers. His current research interests lie in the areas of: planning and scheduling, machine learning, operations research, and decision theory and he has authored numerous publications in these areas.

Helen Mortensen is a Member of Technical Staff in the Planetary Applications Group, Image Processing and Applications Section at the Jet Propulsion Laboratory, California Institute of Technology where she is Multimission Image Processing Subsystem (MIPS) Cognizant Operations Engineer for the Galileo Project which involves automating the Galileo systematic processes such as archiving data to CD-ROMs and producing hardcopy photoproducts. Ms. Mortensen has over 10 years of experience in image processing, her previous duties include software programming for the Multimission Image Processing Laboratory, image processing for the Voyager Uranus and Neptune encounters. She was involved in producing animations such as "Miranda the Movie", "Triton the Movie" and "The Galileo Earth/Moon Conjunction Movie". She was responsible for all the systematic processing for the Neptune encounter which was completed on time and with fewer people due to procedure automation. She holds a B.A. in Computer Science from the University of California at Santa Barbara. Ms. Mortensen's current interests lie in the area of image processing, and intelligent tools for automating image processing tasks.