Vision-assisted modeling for model-based video coding

by

Shawn C. Becker

Thesis Proposal for the degree of Doctor of Philosophy
at the
Massachusetts Institute of Technology

Thesis Advisor:

V. Michael Bove Jr.
Associate Professor
Program in Media Arts and Sciences

Thesis Readers:

Alex P. Pentland
Associate Professor of Computers, Communication and Design Technology
Program in Media Arts and Sciences

Seth Teller
Assistant Professor
Department of Electrical Engineering and Computer Science

Stephen A. Benton
Chairman, Departmental Committee on Graduate Students
Program in Media Arts and Sciences

Abstract

The objective of this thesis is to develop algorithms and software tools for acquiring a structured scene representation from real imagery which can be used for 2-D video coding as well as for interactive 3-D presentation.

A new approach will be introduced that recovers visible 3-D structure and texture from one or more partially overlapping uncalibrated views of a scene composed of straight edges. The approach will exploit CAD-like geometric properties which are invariant under perspective projection and are readily identifiable among detectable features in images of man-made scenes (i.e. scenes that contain right angles, parallel edges and planar surfaces). In certain cases, such knowledge of 3-D geometric relationships among detected 2-D features allows recovery of scene structure, intrinsic camera parameters, and camera rotation/position from even a single view.

Model-based representations of video offer the promise of ultra-low bandwidths by exploiting the redundancy of information contained in sequences which visualize what is largely a static 3-D world. An algorithm will be proposed that uses a previously recovered estimate of a scene's 3-D structure and texture for the purpose of model-based video coding. Camera focal length and pose will be determined for each frame in a video sequence taken in that known scene. Unmodeled elements (e.g. actors, lighting effects, and noise) will be detected and described in separate compensating signals.

Contents

1 Introduction

A great deal of information about the original camera geometry, scene structure and texture can be gleaned from even a small number of uncalibrated views provided the scene contains a sufficient amount of structural information. Certain CAD-like geometric relationships that are invariant under perspective projection and normal lens distortion are available in man-made scenes and can be readily identified. By exploiting these properties among critical sets of 2-D features this thesis proposes to find a coarse estimate for lens distortion, center of projection, camera pose and scene geometry. A refinement stage then uses all feature correspondences from multiple views to refine camera and scene parameters while maintaining known geometric constraints. Image textures from multiple views are registered into common surface coordinates. For surfaces known to have off plane deviations multi-baseline stereo methods are applied to solve local surface disparity. Surface textures are then merged according image resolution and visibility.

If matching and property assumptions of features are correct, if detector noise is reasonable, and if textures are invariant over all views, the final result is a fairly photorealistic textured static 3-D scene model which is true to all views. While this model has several useful applications, model-based video coding is of particular interest. Using the model as prior scene knowledge, we can attempt to recognize novel views of that scene. This task of *pose-determination* can be applied to each frame in a video sequence. Existing approaches to pose-determination rely solely upon having known 3-D edge features, and strive to find corresponding observable 2-D line features. The fact that we have an *a-priori* scene model consisting of both geometry and texture, rather than geometry alone, should improve the accuracy and convergence of pose-determination even in the presence of significant levels of noise (due to modeling error) and clutter (e.g. dynamic unmodeled elements like people).

If the difference between real and predicted sequences is appreciable, it may be possible to segment unmodeled elements into coherent image regions (i.e. 2-D sprites) corresponding to actor extent as viewed from the camera. These 2-D actor sprites can be assigned arbitrary but meaningful 3-D attributes such as having planar orientation normal to the viewing camera and parallel to the scene's vertical axis. If contact between 2-D sprites and any surface of the scene can be determined (e.g. the bottom of the sprite touches the ground plane) or occlusion with known objects can be determined (e.g. the sprite is behind the table) then the sprite can also be given appropriate 3-D position and scale. We might also be able to identify and model other sources for appreciable error, such as environmental lighting changes, shadows, and noise.

1.1 Issues

1.1.1 Motion vision

Existing motion vision approaches like *structure-from-motion* and *shape-from-stereo* use signal processing techniques to detect and match points or lines among several views. For example, optical flow over many views can be used used to recover qualitative measures of camera-relative motion and depth [46, 48, 49, 50]. Multi-baseline-stereo approaches [33] use optical flow or matched features over views with known motion to recover scene structure and texture. Structure-from-motion approaches[24, 25] use large numbers of automatically detected and tracked features over almost-calibrated views[1] to recover camera-relative motion and depth.

1.1.2 Modeling

At the other end of the scale, CAD approaches to scene modeling use critically small sets of features carefully specified with manual intervention. For example, the scene modeling approaches used to create environements for visual simulation [100, 104] use known cameras and scene geometry found from architectural plans or physical measurements. In [90] the user manually corresponds user defined image lines to 3-D edges of manually created sets of parametric volumes which are then automatically adjusted to minimize observed deviation from pre-calibrated cameras. Other systems, like [107], use manually specified point and line features with calibrated cameras at roughly known camera positions.

1.1.3 Shape from wireframes and contours

Another group of approaches, use *shape-from-wireframe* and *shape-from-contours* methods which attempt to use features with known properties to recover object structure or shape. Shape-from-wireframe algorithms aim to recover 3-D geometry of a polyhedral scene (e.g. "blocks world" or "LEGO") from a single line drawing. Initial work [95, 97, 96] focussed on the problem of partitioning wireframe scenes into separate trihedral polyhedral objects made up of collections of labeled vertices and edges. More recently algorithms have been presented [52, 53] which find 3-D polyhedral interpretations of wireframe drawings that match human observations amazingly well. In the special case of trihedral scenes with known vanishing points [13] show that line detection can be constrained and in some cases labeled automatically.

Object contours (i.e. the locus of visible surface points either tangent to the view or discontinuous in surface curvature) are another type of image cue which convey rich geometric information and are stable with respect to illumination changes. Algorithms are presented [54] to recover a description of generalized cylinders by fitting an observed or hypothesized cross section to an observed set of contours. Such algorithms can be used to reconstruct a number of complex shapes, shapes for which polyhedrals are poor approximations.

1.2 Motivations

Digital cinematography, virtual reality and telepresence applications are increasing the demand for photorealism in synthetic 3-D models. To acheive this type of visual quality, some researchers are investigating the use photographic images of real scenes directly rather than attempting to synthesize geometry and texture [65, 90, 107]. Besides attacking the ill-posed problem of structure recovery, this requires that we also face problems of proper registration and merging of texture samples from multiple views. Accuracy and photographic realism is critical, especially if the model is to be used for other applications like model-based video coding where we hope to use a scene model as a stable representation of a scene over multiple views and lighting conditions.

Excellent image-based reconstruction results have been obtained using structure-from-motion techniques. However, since state of the art techniques for feature detection and tracking are far

[1] Where all intrinsic parameters except focal length are known.

from perfect, some sort of human intervention is typically required at that stage to get quality results. In light of the fact that human intervention is often used anyway, existing techniques suffer due to their modeling simplicity since they do not take advantage of powerful geometric properties readily available in many images.

Existing image-based CAD modeling approaches have also given excellent results. However these approaches typically require an initial topologically complete and dimensionally approximate 3-D solution to start with. A faster and simpler approach might be to specify structural constraints among features in a a purely 2-D environment. Existing approaches also impose restrictive viewing conditions by requiring the use of pre-calibrated cameras so the image acquisition stage must be a planned event. It would be more convenient if we could simply model a scene given an existing still image or a video sequence taken from a totally unknown camera.

Recent developments indicate that shape-from-wireframe and shape-from-contour approaches can contribute to the reconstruction problem. These methods can be automated and parallelized and they also offer a good balance between the two extremes of vision-based reconstruction and manual 3-D specification. They allow richer type of geometric reconstruction than point or line correspondences alone and because they contain more structural information they can be used to more strongly constrain equations for solving structure from images.

Another motivating factor is to contribute to the goal of realizable 3-D video capture from monocular 2-D cameras. Since acquisition of photorealistically textured 3-D models directly from images or film is difficult, few techniques exist which attempt to exploit that knowledge for model-based video coding. With the help of an able thesis committee, I hope to be able to make new contributions in this area.

1.3 Thesis Committee

Besides the fact that the members of this proposed thesis committee share my enthusiasum for this topic, they were chosen because of the particular interests and expertise that each can bring to this effort.

Professor Bove has experience in the topics of multi-sensor fusion, digital representations for video, and hardware systems for video coding/decoding. His knowledge and resources will make it possible to develop methods for 3-D reconstruction from real imagery of man-made scenes which can be used for low-bandwidth 2-D video coding. The results of this research will also contribute to previous work he's done with interactive 3-D structured video.

Professor Teller has experience in designing efficient representations for storage, retrieval and rendering of large complex graphical databases. Since texture rendering is an inherent part of this approach to video coding and decoding, his knowledge may help find methods to animate photorealistically textured 3-D scenes with incremental changes in camera position. His shared interest in acquiring realistic scenes for visual simulation also make him a natural choice as a committee member.

Professor Pentland has experience in photogrammetric methods for scene recovery as well as dynamic scene understanding. His suggestions will be valuable for developing methods to recover a representation for scene structure which is optimally stable over multiple views, to use that model for understand novel views of that scene, and to segment image regions containing human figures when the background is known. Results of this research could be used to provide photorealistic 3-D scenes for immersive virtual environments.

1.4 Goals and contributions

The main objective of this thesis is to develop algorithms and software tools for acquiring a structured scene representation from real imagery which is amenable for 2-D video coding as well as for interactive 3-D presentation. New and significant contributions, include:

- A new method of camera calibration which uses the vanishing points of three mutually orthogonal sets of parallel lines to solve lens distortion, center of projection and camera pose simultaneously.
- A new structure and motion algorithm which besides using point and line correspondences among images, additionally exploits geometric constraints like coplanarity, parallelism and perpendicularity which can be defined with a modicum of user intervention.
- A new algorithm for refining camera pose given approximate camera pose in a scene with known texture and geometry with straight edges.
- A new algorithm for actor segmentation given known camera pose in a known static scene.
- An improved method for model-based video coding which is amenable to interactive 3-D manipulation and presentation, and describes a video sequence in terms of a textured static 3-D scene model, changing camera pose, dynamic 2-D actor sprites, and a residual error signal.

2 Background

To familiarizes the reader with issues of image-based scene reconstruction the following sections review issues in camera calibration, structure-from-motion algorithms and pose-determination. We will also review current work in model-based video coding and virtual reality over the World Wide Web; two areas that could benefit from the availability of highly realistic and accurate models recovered from new or preexisting imagery.

2.1 Camera calibration

Images created with a compound lens system are typically modeled by a pin-hole projection equation which describes how 3-D elements in the scene are projected onto 2-D positions in a frame buffer. This non-linear projection equation can be described as a set of intrinsic and extrinsic camera parameters. Intrinsic parameters include radial and decentering (also known as tangential or prismatic) lens distortion terms, horizontal pixel scale, pixel shear, principal point (also known as image center or optical center) and effective focal length. Extrinsic parameters (also known as camera motion or pose) describe the camera's position and rotation with respect to some reference frame, either some object in the scene, one of the cameras, or the camera's position at some instance in time. Scene structure refers to the 3-D geometry of visible points, lines and surfaces. *Camera calibration* algorithms, use known (or partially known) scene structure to determine a camera's intrinsic parameters.

2.1.1 Calibration using direct measurements

Intrinsic and extrinsic camera parameters can be obtained from physical measurements of camera itself or by using three-dimensional coordinates of known world points and their corresponding 2-D projections in the image. Typical approaches [1, 2, 3, 4, 5, 6, 7] employ specially constructed calibration templates (usually a planar grid) with a large number of precisely positioned fiducial points or lines. Due to the non-linear nature of the projection equation, these calibration approaches use interative nonlinear optimization techniques, and thus are dependent upon a reasonable initial estimate. However, by using large numbers of precisely known template features very accurately calibration parameters can be found.

If distortion can safely be considered negligible, Roberts [93] shows that the remaining parameters can be combined to form a single homogeneous 3 by 4 matrix which can be solved given 6 or more known world points and their 2-D projections in two images. To determine of the position of the camera itself relative to known object points, Ganapathy [94] shows how this 3 by 4 matrix can be inverted into its constituent intrinsic and extrinsic parameters.

2.1.2 Affine calibration

This matrix can also be used with corresponding pairs of 2-D points to recover a more qualitative description of scene structure. As described by Faugeras [9, 10] and others [11] given 4 non-planar point correspondences, point-wise scene structure can be recovered up to an arbitrary affine transformation relative to the positions of those four points in three-dimensional space. This formulation completely evades the effects of camera calibration by recovering only relative information at the expense of losing metric. The subsequent gain is that these approaches are more stable, simpler and better behaved. Such an approach is appropriate when accurate metric information is not required, such as in robotic systems using binocular stereo vision where relative positioning is all that is needed.

2.1.3 Calibration using three-point perspective

Another class of techniques lets us make assumptions about geometric shapes of elements or relationships among elements in the scene. This offers advantages over approaches using direct scene measurements since we can recover geometry to an arbitrary scale factor from a single image. For example it is well known that the projection of a set of parallel lines in a scene form a set of lines in an image which meet at a common *vanishing point*. If a scene contains three mutually orthogonal sets of parallel lines (i.e. a rectangular solid or right parallelipiped) an image of that scene can contain one, two or three vanishing points. Interior and exterior views of man-made buildings are prime examples of scenes containing these structures. If an image contains three vanishing points, there exist several properties of projective geometry which are particularly useful for camera calibration.

- the 3 vanishing points of the 3 sets of parallel lines form a triangle whose orthocentre is the principal point or image center
- the vector from the center of projection to a vanishing point is parallel to the set of lines in the scene.

- the vanishing points of all lines that lie in a 3-D plane in the scene lie on a common line in the image which is called the vanishing line for that plane
- the position of the 3 vanishing points change only due to camera rotation not camera translation.

Caprile and Torre [12] and Wang and Tsai [14] exploit these properties to determine principal point and focal length from the vanishing points in a view of a cube. These algorithms assume known or negligable lens distortion. Since they use so few features, these calibration approaches are quite sensitive to detection errors.

While previous examples use 3-point perspective to calibrate cameras using rectangular solids of *known dimensions*, Becker and Bove [16] show how one or more perspective projections of a right parallelipiped with 3-point perspective can be used to recover rectangular object dimensions as well as intrinsic and extrinsic parameters. Camera position and object size are solved to an arbitrary scale factor and horizontal pixel scale cannot be recovered and must be assumed unitary. No numerical results of estimation errors are presented but it was found that mean estimation error of center of projection (and consequently estimations for camera pose and scene structure) increases as the object's angle of view decreases, as mean line segment length decreases, or as true focal length increases.

Robust algorithms for finding vanishing points from a sets of image lines are presented in [17, 19, 18]. Error analysis of vanishing point estimation is discussed in [20, 21, 22].

The disadvantage of 3-point perspective approaches to camera calibration is the requirement that sufficient rectangular structure is visible in the scene. The clear advantage is that the majority of intrinsic and extrinsic parameters can be accurately recovered without requiring explicit measurements, but simply by taking advantage of geometric properties of objects visible in a single image.

2.1.4 Plumb-line techniques

Another example of an approach that takes advantage of geometric properties in scenes to solve intrinsic camera parameters is Brown's [23] *plumb-line* technique for solving lens distortion. Given a single image of a scene of parallel edges, the effects of lens distortion can be corrected by iteratively finding the distortion parameters that minimize curvature among points in each line in the distorted image. This approach to solving lens distortion differs from template-based calibration techniques in that no knowledge of world points is required. Rather, linear relationships between points (i.e. straightness) is all that is known.

Becker and Bove [16] improve upon the plumb-line technique by exploiting the vanishing point property of parallel lines as well as Brown's straightness constraint. They present an iterative technique for finding the distortion parameters which minimize the vanishing point dispersion derived using Kanatani's [20] statistical foundations. Using this formulation, lines which are parallel in the scene are constrained to be straight as well as share a common vanishing point in the image.

2.1.5 Projections of spheres

Assuming negligible radial lens distortion and a scene with two or more spheres, Stein [8] solves principal point, effective focal length and horizontal pixel scale by analyzing each sphere's elliptical projection on the image plane. This approach is sensitive to distortion since when even small amounts of radial lens distortion are present, the affects of focal length and radial distortion are impossible to differentiate.

2.1.6 Calibration using properties : Overview

These approaches require that the scene contain structures with certain geometric properties that can be exploited to determine intrinsic (and sometimes explicit) camera parameters. In a sense, they require some sort of qualitative and relative knowledge of world points rather than requiring explicit positional knowledge of world points. This fact suggests that approaches that recover non-metric structure to an arbitrary affine scale could be modified to take advantage of certain easily defined properties, such as orthogonality, or equi-distance to attain metric.

2.1.7 Calibration using unknown world points

Another approach that is gathering interest is to use large numbers of features and/or large numbers of frames to overconstrain the non-linear observation equations used in structure and motion determination (which we review in greater detail in section 2.2). If these equations are sufficiently overconstrained, intrinsic parameters (like focal length) can be solved along with extrinsic camera parameters and scene structure simultaneously as shown by Azarbayejani and Pentland [24, 26].

2.2 Structure-from-motion algorithms

The recovery of scene structure from 2-D images alone is a convenient and powerful means of quickly acquiring knowledge about a scene. Classic examples of approaches to visual scene reconstruction consist of a variety of so called "shape-from-X" algorithms where X might be shading, texture, focus, contour, wireframe, stereo, multi-baseline, or known motion. Structure-from-motion approaches, use the visible artifacts of "motion parallax" over two or more views to recover unknown scene structure as well as unknown camera motion. These approaches typically assume that intrinsic camera parameters have already been found by calibration, though some so-called "self-calibrating" techniques attempt to recover some intrinsic parameters as well, as mentioned in section 2.1.7. Feature correspondences are used to determine the 6 extrinsic parameters that describe the pose (i.e. position and rotation) of one camera relative to another up to an unknown scale in position. Having determined relative pose (or relative orientation), the 3-D position of features relative to one camera can be found by triangulation.

2.2.1 2-D to 2-D point correspondences

Examples of structure-from-motion techniques which use critical sets of matching 2-D points among two or more images include [31, 28, 29, 30]. To combat difficulties presented with trying to solve the non-linear projection equation with noisy or erroneous feature matches, some researchers are using long image sequences to overconstrain the problem. Several approaches have been presented [39, 40, 24] which use incremental estimation frameworks based on the Extended Kalman Filter or EKF [38]. Others [32, 27] use batch estimation frameworks, having higher computational overhead but potentially better accuracy since they avoid the linearizing assumptions inherent in the EKF.

2.2.2 2-D to 2-D planar points and orthogonal corners

If these matched 2-D points in multiple images have some known structure, then this structure can be exploited to generate additional equations and this results in requiring fewer points for finding a solution for 3-D point position and camera pose. Several approaches [43, 42, 41] show how 4 or more matching points known to be coplanar provide solutions for incremental camera rotation, plane orientation and translation to an unknown scale factor. Others [15] use orthogonal corners as features, that is points that mark the intersection of 3 mutually orthogonal edges. Given a calibrated camera, one orthogonal corner and two point or line correspondences are all that are needed to determine motion and structure uniquely.

Given a pair of uncalibrated images, we would not expect a structure-from-motion algorithm to automatically recognize such high level properties of points. We include them, however, because *their use in conjunction with an interface tool* is of exceptional value.

2.2.3 2-D to 2-D line correspondences

Straight line features offer definite advantages over point features in that they are prominent in man-made scenes and they are easier to detect and track. Straight line features provide useful structural information (as shown in section 2.1) and since they have more image support they can be localized more accurately. While these properties make lines advantageous for use in structure-from-motion algorithms they require more views and more correspondences than points. As described in [28, 34, 37] at least 3 views of 6 or more line correspondences, are required as long as the lines are not arranged in certain degenerate configurations.

2.2.4 Optical flow and spatiotemporal derivatives

Optical flow can be interpreted as the field of 2-D image motion caused by 3-D motion of a rigid object with respect to an imaging sensor. Theoretically, it can be used to recover relative motion and depth [44] but its use for anything but a qualitative measure of structure and motion in real image sequences is questioned by [45]. Structure-from-motion algorithms using optical flow and spatiotemporal brightness gradients directly have been presented by [46] and [48, 49, 50].

2.3 Pose-determination from known structure

Suppose we use camera calibration and structure-from-motion techniques to recover an accurate photorealistic model of a scene from a few stills or key frames of video. In this section we review how to automatically determine camera pose given a new view of that scene.

Pose-determination is usually considered a subset of the problem of object recognition, where given a database of objects the task is to recognize the object in a real image. A typical strategy for object recognition is to take each object and find the transformation (i.e. camera pose) which aligns it to best fit some set of features in the 2-D image. If an object's aligned fit acheives some threshold that set of features is identified as that object with an associated pose.

2.3.1 Critical sets of 2-D to 3-D matches for pose-determination

Pose-determination techniques use the same photogrammetric methods as those mentioned in section 2.1.1 except that they are optimized to take advantage of only critically small sets of matches which are typically available in uncontrolled situations.

For the case of point features, Ullman [84] shows how 3 pairs of 2-D to 3-D point matches can be used to find a unique transformation describing weak perspective (i.e. orthography and scale factor). For normal perspective, 3 point pairs can be used to form a fourth-degree polynomial in one of the unknowns, thus resulting in as many as 4 possible solutions for pose. Haralick [56] gives a good review of these techniques and their relative numerical stability. If 4 or 5 point correspondences are available, either a linear least-squares solution can be found [28] or Fischler and Bolle's RANSAC method [55] can be used to find the solution that best describes all observations. Ganapathy [94] presents a method that in general produce a unique solution given 6 or more point correspondences as long as the points are not colinear on the image plane.

Use of line correspondences may be preferred over points because of their relative ease of detection and tracking. As shown by Liu, Huang and Faugeras [57], EKF [38] can be used to first find camera rotation given 3 line correspondences, and then 3 additional line correspondences are sufficient to determine camera translation. To explain why even non-critical sets (i.e. more than the theoretic minimum) of line correspondences do not always give robust results, Navab and Faugeras [34] show that for certain degenerate configurations no unique solution for pose can be found. Kumar and Hanson [58] present a RANSAC-based iterative technique which minimizes error due to rotation and translation simultaneously, and claim its superiority to techniques that solve for rotation and translation separately.

If individual matches between 2-D and 3-D point features can't be made, but sets of points can, Nagao and Grimson [79, 80] show how 2nd order statistics of spatial distributions of matched clusters of point features can be used to solve camera pose under weak perspective projection up to an unknown image plane rotation. Nagao and Horn [81] present a similar technique that uses direct moments of image intensities and intensity gradients.

2.3.2 Strategies for feature-based pose-determination

Approaches for determining arbitrary pose given a hypothesized model can be categorized into 3 groups. Pose search methods [37, 74] attempt to test the space of all poses to find the one that best fits the model to the observed image. Pose clustering methods [85] use techniques like the generalized Hough transform [17] to accumulate evidence for possible coordinate transformations from all possible matches. And alignment methods [84, 55, 78] compute pose from all possible matches of critical sets and then see if any of the remaining model features are supported by image features.

If we consider that four 2-D to 3-D matches are needed to find a unique camera pose, in the general perspective case, then straight out testing of all hypothetical pairs of model and image features has 8th order cost in the number of features. Region segmentation and region matching effectively groups features into smaller sets for comparisons. This brings the testing cost down only quadratic order in the number of regions. Approaches for color histogram-based region segmentation and matching are found in [86, 77]. Texture analysis and synthesis results can also be used for texture-based region segmentation as shown in [88, 36, 89].

Another way to prune the number of hypothesis tests needed for pose determination is consider only those 2-D and 3-D features that have similar identifying characteristics. Properties which are invariant (i.e. consistent) even under large changes in viewing or lighting conditions are particularly useful. Some examples include junction labeling of points [76], color opponency of points and lines [80], as well as color and texture metrics for region matching.

2.3.3 Strategies for iconic-based pose-determination

While feature-based approaches to pose-determination use a difference measure which is based on mis-registration of 2-D point or line features, iconic-based approaches can be used instead which compare image depths or image intensities directly. Holtzman's [74] uses a straight forward pose search strategy using textures. Viola [82] iteratively finds the pose parameters which maximize the mutual information (i.e. functional *consistency*) between pixel depths of the rendered model and observed image intensities.

2.3.4 Pose-determination : Overview

Feature-based methods for pose-determination are combinatorically expensive but robust in the presence of clutter, noise and occlusion. Iconic-based methods for pose-determination require a full texture rendering at each iteration and thus are extremely processor intensive. However, they neatly side-step the inherent problems of feature detection and matching and thus may be more applicable to parallel architectures that implement both 3-D texture rendering and 2-D image processing pipelines. For these reasons, iconic approaches also lend themselves nicely to coarse-to-fine refinement of pose over different scales. The relative strengths of feature- and iconic-based strategies suggest that a coarse to fine strategy using first iconic and then feature matching might be useful.

2.4 Model-based video coding

Communicating a higher-level model of the image than pixels is an active area of research, particularly for the application of low-bandwidth video transmission for teleconferencing. The basic idea is to have the transmitter and receiver agree on the basic model for the image. The transmitter then sends parameters to manipulate this model in lieu of picture elements themselves. These types of model-based decoders are essentially computer graphics rendering programs which trade generality for extreme efficiency in its restricted domain.

We can get more generality if the decoder can accept any model which is then manipulated. In this section we first study existing work done in model-based video coding of humans and then review video coding of static scenes. We then consider how structure-from-motion, pose determination and figure finding techniques can be used to get a combined description of humans and static scene so we can enter the next stage: dynamic 3-D video coding.

2.4.1 Model-based coding of human figures

The area of model-based video coding of human figures is a hot area of debate as forums attempt to agree upon standards for low-bitrate video transmission formats such as MPEG-4 [103]. What could be the first application of model-based video coding of humans was presented in 1983 by Forchheimer and Fahlander [61] using animation to describe facial image sequences. They introduced the notion of giving the sender and receiver a shared set of models of objects seen in common image sequences (i.e. a human face and shoulders on a static background). For the special case of head-and-shoulder scenes, Li, Roivainen and Forchheimer [62] suggest that motion parameters for head-and-shoulder scenes be separated into global rigid motion and local deformations due to facial expressions. Essa's work [63] represents the state of the art in compact model-based representations for facial expression. To make facial expression appear more life-like, Provine and Bruton [83] explore psychovisual models that "trick" the eye just as psychoacoustic models are used to "trick" the ear in audio coding.

2.4.2 Detecting static content in a video sequence

Other research efforts aim to recover information about the entire scene, not just a moving human figure, from video sequences. Contributing to the goal of determining static visual content from a set of monocular views with a moving camera, Adelson and Bergen [64] formalized the concept of 2-D visibility over the entire sphere of possible directions as the plenoptic function. If we are given images from a moving camera, and the motion is purely rotational (i.e. constrained to rotate about the exact center of projection) with changing focal length (i.e. zoom), the set of images can be registered and "mosaiced" onto a common sphere of possible directions.

2.4.3 Static plenoptic function from pure rotation

Methods for finding transformations that register images into the plenoptic function use either point correspondences (or equivalently optical flow) or are iconic in nature and thus require no explicit point matches. Aizawa [59] presents an iterative technique for finding pure rotation, focal length and zoom given point correspondences in a pair of views. McMillan and Bishop [65] describe transformations needed to project images onto a common cylinder, claiming to be able to solve 5 intrinsic parameters as well as 3 parameters of pure rotation from 12 or more feature correspondences. Mann and Picard [66] present a feature-less iconic approach for estimating the 8-parameter perspective transformation used to reproject and register multiple images onto a common plane via "video-orbits" for image mosaicing.

If the focal length is considered infinite (i.e. having viewing conditions of pure orthography), a 6-parameter affine transformation describing image flow over an entire region can be found as shown. This affine transformation is used by several model-based video coding algorithms for image mosaicing and resolution enhancement [67, 69, 70]. If rotation, scale and shear are fixed, then the transformation simplifies to the purely 2-D translational model such as the block-wise translation model used for MPEG-I and MPEG-II digital video formats.

2.4.4 Moving toward a static 3-D scene model from general translation

Algorithms have been developed which aim to interpret the field of visual motion as distinct objects moving relative to the camera. Attempts have been made [68, 69] to iteratively classify optical flow into regions which fit a common set of affine motion parameters. In this way, they effectively model an image sequence as a set of planar textured "layers" being viewed under orthography. Kermode [70] developed a hybrid video coder that uses similar principles but adds capabilities for image mosaicing and resolution enhancement and works within the current

MPEG digital video coding framework. Adiv [46] attempts to fit optical flow to locally planar patches and then does local hypothesis testing to aggregate patches that have compatible motion parameters (i.e. the 8-parameter planar perspective transformation used by [43]). Adiv's result can be considered to similar to [68] except that a 3-D perspective camera model was assumed.

Here we see a blurring between efforts of video coding and total 3-D scene reconstruction. As pointed out by [45] structure-from-motion algorithms are ill-suited for general video coding because of the difficulties of feature detection and tracking due to the aperture problem (which is that image flow can only be resolved in the direction of available texture detail). It is also difficult to automatically cluster optical flow vectors into regions of similar planar perspective flow due to its non-linear form and sensitivity to noise in 2nd order spatiotemporal derivatives. In contrast, affine models, effectively model optical flow over a region as flow caused by viewing a moving planar texture under orthography. Although the affine model is only an approximation to true perspective flow, it is in practice more tractable because it has a linear solution which uses only 1st order spatiotemporal brightness derivatives.

2.4.5 Dimensionalization

Holtzman [74] sidesteps the optical flow perspective segmentation problem for video-coding by allowing partially known scene geometry. In an initial frame, 3-D planar polygons describing key surfaces in the scene are manually placed via a CAD system relative to the initial camera's origin with known focal length. In this process, which he calls "dimensionalization", the user aligns known world geometry to observed image features and serves to bootstrap the process for coding video sequences created with a moving camera in a static scene. Relative pose for the next frame is found by searching the 6-parameter space to find the pose that minimizes pixelwise difference between the next image and textures from previous image transformed according to plane geometry and new camera pose. In this way, Holtzman showed that the concept of "dimensionalization" offered practical advantages over other methods of video coding in that it made 3-D model-based video coding practical if allowed the initial startup cost of manually specifying the 3-D scene.

2.5 ISIS : a custom delivery mechanism for model-based video

Isis is a scheme-like interpretive programming language written by Stefan Agamanolis [73] which can be used as a stand-alone programming environment as well as interface language to packages of C functions that do specialized operations from networking and file access to synchronized processing of graphics and sound.

Agamanolis uses this framework to create what we will refer to as the Interactive Model-Based Video Decoder (or IMBVD). This decoder uses a format for model-based video which has objects and a script. Objects have 2-D, 2.5-D and 3-D geometry, audio, and video properties. The script specifies how object properties change as a result of user manipulation and with the persistant passage of time. If run on a platform with sufficient processing power, the IMBVD can handle real-time audio and video rendering/composition as the user interactively manipulates camera parameters and global valuators. Individual objects have geometry, video texture and audio and as the point of view changes stereo sound and video texture for each object are rendered at the appropriate location relative to the viewer. The IMBVD also handles special video effects like filtering, warping and chroma-keying as well as audio effects like echo and reverbration.

Though Isis can work on any platform, a high-performance implementation of the IMBVD exists on the Cheops [114] platform, the Media Lab's modular platform that performs computationally intensive stream operations in parallel in specialized hardware. Since the architecture is optimized for real-time 2-D and 2.5-D alpha-depth-buffered composition but not for 3-D texture mapped rendering, textured 3-D objects are pre-rendered as 2.5-D objects (i.e. color + alpha + depth) from an arbitary set of scripted views. During playback the IMBVD on Cheops thus supports interactive viewing within that subset of views.

2.6 VRML : another possible delivery mechanism for model-based video

Of late, the increasing availability of graphics rendering hardware, explosive interest of the Web, and proliferation of the Virtual Reality Modeling Language (VRML) as a standard 3-D scene description language has ignited a flurry of interest in creating engaging environments. In 1994 David Raggett proposed the concept of using a new markup language to extend the World Wide Web to support platform independent virtual reality [102]. He envisioned a high level language describing VR environments as compositions of logical elements. Geometry and complex behaviors of these elements could be described at different levels of detail and retrievable as shared resources from the network via the universal reference locator (URL) scheme. In time, this concept become codified into a specification that has paved the way for VRML browsers on many platforms.

In an effort to lay the groundwork for VRML, the initial specification described little more than static geometry and texture in scenes [105]. From its initial conception, however, the formative goal, has been to support dynamic properties of elements such as object behavior, audio and video, as outlined by Pesce [106]. Now as academic and commercial interests for virtual reality applications on the Internet gain momentum, VRML designers are rapidly adding capabilities to encompass those goals. Recently[2], competing specifications for version 2.0 have been proposed by Microsoft [115] and Silicon Graphics [116]. Both proposals support scripted event-driven behaviors for agents with graphic and audio properties, similar to 2-D capabilities already supported by Isis. Eventually, VRML agents may also be able to support data streams to deliver live actor geometry and MPEG-II video textures. VRML may eventually become a practical delivery mechanism for model-based video over the Internet, but until then, ISIS is the clear viewer of choice.

[2]Friday, February 23, 1996

3 Research Proposal

3.1 Overview

One of the Media Laboratory's hallmark visions has been to support interactive 3-D visualization of real events. A prime example has been to be able to watch a football game from an arbitrary point of view, even from the point of view of the football. Initially, such a task might seem physically impossible, but telepresence applications such as this might actually be tractable if a photorealistic and accurate model of the static 3-D scene can be constructed beforehand.

A computer graphics model of extreme accuracy and detail can be constructed by hand using physical measurements of the scene and traditional CAD modeling techniques. However, this approach is labor intensive, time consuming, and requires that the scene be physically accessable. Self-calibrating structure-from-motion algorithms can be used to get a scene model directly from an image sequence, but this approach requires many feature correspondences over a long sequence of images, and may still not provide the required level of detail and structural accuracy.

Some in the vision community [37] have suggested that the accuracy and robustness of structure-from-motion algorithms can be improved if we go beyond simple feature correspondence and attempt to exploit other geometric relationships among features. This thesis work attempts to satisfy this need. By allowing basic CAD-like constraints (e.g. coplanarity, parallelism, and perpendicularity) to be defined on image features directly, we may be able to get useful accuracy and detail from fewer feature correspondences in fewer images.

The primary goal of this thesis is to develop a complete system for 3-D model-based video encoding. The **initial task**, which is very near completion, is to develop a vision-assisted approach for modeling a static man-made scene given one or more overlapping images. Section 3.2 details a new *shape-and-texture-from-wireframe* approach for recovering camera and scene geometry. This approach has been implemented as a program called `sceneAnalyze` which takes a 2-D scene description (an organized collection of lines in one or more images) and outputs a structured scene description in the form of 3-D edges and textured 3-D polygons in formats which are readable by Isis or VRML viewers.

Section 3.3 reviews two completed modeling programs: a primitive X11-based system developed by the author and a more user friendly Motif/OpenInventor-based system developed by Eugene Lin. Both modeling systems output a 2-D scene description ready for batch processing by `sceneAnalyze`.

The **final task**, which has yet to be done, is to develop an approach for using this sort of photorealistic model for motion vision analysis and actor segmentation in real video. Section 3.4 will propose an approach for decomposing video of a known scene into pose parameters, 2-D actor billboard sprites (with position and planar orientation) and a 2-D residual error signal. For convenience, these video components will be output in a format compatible with existing Isis viewers.

3.2 Work completed : A vision-assisted approach to scene modeling

This section introduces an approach to vision-assisted scene modeling which has already been implemented. The first 6 subsections discuss details of the information gathering process consists of different stages requiring varying degrees of manual intervention: image capture, line detection, specifying parallel constraints, specifying an arbitrary world frame and scale, specifying coplanar constraints and optional specification of additional line correspondences.

This information, which is largely an organized collection of lines and line properties in one or more images, makes up a 2-D scene description. This 2-D scene description is used by a shape-and-texture-from-wireframe algorithm to recover a textured 3-D description of the scene. The shape-and-texture-from-wireframe algorithm consists of 2 automated batch processes: structure-and-motion-recovery described in subsections 3.2.7 and multi-view texture recovery described in subsection 3.2.8.

3.2.1 Image capture

One or more *images* of any size, orientation, focal length, distortion, or exposure can be used. Still images can be digitally scanned to arbitrary resolution using a flatbed image scanner or individual frames of video can be digitized using standard video capture hardware.

3.2.2 Line detection

The Canny [71] edge detection algorithm is applied to each image after which a line detection algorithm [72] is used to find and traverse continuous chains of edge pixels which are broken into piece-wise linear segments. Each *line* in an image is given its own *edge* attribute. After structure-and-motion-recovery, members of the edge class describe 3-D edge position in the scene for matched 2-D lines in one or more views[3].

3.2.3 Specifying parallel constraints

All 2-D lines which come from a set of parallel 3-D edges in the scene, regardless of the image they are in, need to be assigned the same *direction* attribute. A direction is a class which after structure-and-motion-recovery describes the world-relative direction of some parallel set of 3-D edges in the scene[4].

Automatic line classification

Under perspective projection, 3-D edges which are parallel in the scene give rise to 2-D image lines which share a vanishing point. In some cases, lines which share the same vanishing point can be automatically grouped into the same direction. We can represent 2-D lines in the image as N-vectors [20] or epipolar normals on the Gaussian sphere of unit directions whose center is some approximate center of projection[5]. Each set of lines that comes from a set of parallel 3-D edges in the scene have N-vectors which form a portion of a great circle on the Gaussian sphere. The great circle lies in a plane, the normal of which points in the direction of that set's vanishing point in the image. To automatically classify lines according into some number of directions,

[3]Explicit line matching among views is allowed but not required since matching is done implicitly for certain lines as explained in subsection 3.2.5

[4]To aid in visual verification, the direction class can also describe graphical attributes of all associated lines (e.g. line color, line thickness and dash style).

[5]The exact position of the center of projection with respect to the image need not be known for direction-based line classification, but will be estimated later during camera calibration

any clustering method, like ISODATA or K-means, can be used to cluster N-vectors into that number of planar great circles. The major benefit of doing the classification on N-vectors is that clustering sensitivity is not related to vanishing point position.

Manual direction correspondence among views

At or near the horizon line for a plane, edges in that plane which have different directions in the scene produce colinear lines in the image. As a result, these lines cannot be automatically classified into distinct direction sets and must be reclassified *manually* via a graphical user interface. User intervention is also required to correspond entire direction sets among views.

3.2.4 Specifying an arbitrary world frame and scale

So that the structure-and-motion-recovery stage can establish a coordinate frame for the scene, at least three non-coplanar directions must be assigned arbitrary world vectors. These directions are now considered to be *known* or *fixed* directions and will form a basis that establishes a rigid coordinate frame for all 3-D scene elements and cameras.

To establish arbitrary position and scale, the 3-D edge associated with one 2-D line in the scene must be assigned a known 3-D position, direction and length in pixels.

Each image has an associated *camera* which describes the intrinsic parameters and pose of the physical camera from which that image was created. By default each camera is considered to be in the first of 8 octants relative to the scene's coordinate frame[6]. If this is not the case, to help establish the camera's octant, one line of at least two known directions in an image must be given an arrow manually. Line arrows indicate the heading of a line's direction vector as viewed in a particular image[7].

3.2.5 Specifying coplanarity constraints

All 2-D lines which come from 3-D edges which are coplanar, regardless of the image they are in, must be manually assigned to have the same *surface* attribute. Surfaces are infinite 3-D planes which become fixed in the analysis stage[8].

Assigning lines a common surface attribute serves to match sets of lines among multiple views. If a line is assigned two surfaces then that line marks the *hinge* or intersection between two planes. In order for the structure-and-motion-recovery stage to recover surface positions, all surfaces must share some hinge-line with another surface. Scenes with this property are sometimes called "origami" scenes [13].

3.2.6 Optional specification of additional line correspondences

If the two surfaces mentioned above have visible lines in a second image, then the hinge-line in the second image implicitly matches the hinge-line of the first. These implicitly matched

[6]Octant is used to resolve among multiple mirror solutions that occur in a relative orientation formulation.

[7]Alternate methods are to provide the octant number explicity or to provide a very rough estimate of camera pose, but putting arrows on lines is both simpler and provides a useful graphical context which helps the user remember orientation when working with many images with very different poses.

[8]Like directions, a surface can be used to describe graphical attributes of all associated lines (e.g. line color, line thickness and dash style) to aid in visual verification.

hinge-lines among several views are sufficient for scene construction. However, accuracy of the final reconstructed scene will improve as more line correspondences are supplied explicitly.

3.2.7 Structure-and-motion-recovery

Unlike conventional approaches which use simple point or line correspondences from multiple views, we promote the use of a critical set of structured elements which normally **require** manual specification but also allow a degree of camera calibration as well as pose-determination from one or more images. By using features that exploit these added constraints, we claim that solutions for scene structure and camera motion will be more robust, more accurate and in many cases allow recovery of structure from a single view (or equivalently, under conditions of pure rotation).

Requirements for camera calibration:

- two or more lines in each of three non-coplanar directions with known 3-D heading
- exactly one line of known 3-D position, 3-D direction and length

Lens distortion correction
In each image, a batch estimation technique, described in a previous article[16], uses the downhill simplex method[9] to find the distortion parameters which globally minimize the vanishing point dispersion for all lines of the same direction. As described in [16] these optimum parameters are used to undistort image textures via stochastic resampling and to undistort associated point and lines.

Independent view calibration
Image relative center of projection, or COP, is found directly from the positions of vanishing points from lines which are associated with 3 known directions. The COP has three terms, 2 of which describe the camera's *principal point* in pixels and 1 term describing *focal length* in pixels. This calibration approach assumes known horizontal pixel scale and pixel shear.

This method of COP estimation is sensitive to the camera's actual orientation and requires viewing conditions of 3-point perspective (i.e. when all 3 directions produce finite vanishing points on the image plane). If 3-point perspective is not available, a-priori knowledge of one COP term and conditions of 2-point perspective are sufficient to solve the remaining two COP terms. Failing this, knowledge of two COP terms and 1-point perspective with a receding line of known distance are sufficient to determine the single remaining COP term.

Center of projection is typically considered to be positioned along the axis which passes through the geometric center of the image. Although this might be true for the film plane of the physical camera, several factors can make this untrue for the final digitized image: possible horizontal delay in video acquisition hardware, physical misregistration of a photograph or negative on the scanner bed, or uncentered cropping. Consequently, we attempt to solve all 3 terms of center of projection using information contained in the digitized image itself.

[9] a gradient descent method that needs only function evaluations at each iteration not Jacobian or Hessian evaluation[75]

For most computer vision setups, the COP is considered to be the origin of the camera's coordinate frame and a virtual image plane is positioned one focal length in front of the COP toward the scene. In an actual physical camera, however, the COP is an imaginary point lying somewhere along the optical axis inside a series of lenses which is positioned between the scene and a film plane. Mathematically, the two arrangements seem identical and the first simpler virtual framework is usually preferred.

In this thesis, we use the physical model since it more correctly models the physical imaging situation (e.g. we can take advantage of knowledge that the film plane is held rigid while the focal length is changed)[10].

If these intrinsic parameters are physically known to be held constant, very accurate estimates of distortion and center of projection can be made by accumulating evidence over many frames. This allows us to approach the level of accuracy usually only obtained by template-based calibration methods.

Requirements for scene structure and camera pose reconstruction:

- an "origami" scene.
- at least one surface containing a line whose edge has known 3-D position.
- each surface must have 2 or more "hinges" whose 3-D directions are either known, can be recovered from vanishing points, or can be recovered by triangulation from 2 or more views.

Coarse scene structure and camera pose

Coarse scene structure and camera pose are now solved in an iterative stage.

- If an image has enough lines with fixed edges or directions then pose is estimated using weighted least-squares 2-D to 3-D pose-determination techniques (described in section 2.3.1) where weighting is proportional to 2-D line length and inverse projected distance.
- Known pose and coplanarity constraints are used to fix unfixed edges in a weighted least-squares manner by projecting from lines onto fixed surfaces or by intersecting projections from one or more views.
- Fixed edges are used to do a weighted least-squares fit for unfixed surfaces and unfixed directions.
- Continue this process until no new edges, surfaces or cameras have been fixed.

Fine scene structure and camera pose

Keeping scene directions and camera distortions fixed, scene surfaces, camera pose and camera COPs are adjusted in a global refinement stage.

[10] Ali Azarbayejani has recently pointed out that another possible benefit of using the physical model is that it does not under-represent backprojection error as the virtual model seems to do.

- A scene vector is formed which contains all refinable camera and surface parameters using 9 terms for each camera (3 for rotation, 3 for translation and 3 for COP) and 1 term for each surface[11].
- Define a cost function based on sum of hinge-line misregistrations. This is formed by using the current parameterization of surfaces to get new infinite hinge-edges (edges which mark the intersection between pairs of non-parallel surfaces). Infinite hinge-edges are projected via parameterized cameras to their respective image plane and compared to observed lines.
- Use downhill simplex method[75] to find the scene parameter that minimizes this cost function.
- Use final scene parameterization to update cameras and surfaces.
- Using final cameras and surfaces to project observed lines into scene to get final weighted least-squares infinite edges. Then clip infinite edges to max extent of reprojected lines.

3.2.8 Multi-view texture recovery

Having refined camera and surface geometries, image pixels can now be projected onto visible regions of surfaces. By default, each image region has an extent defined by the convex hull of all lines in that image which have been assigned to a particular surface. If the image region should actually be non-convex (such as when a wall has a door frame which should be transparent) the shape of the image region can be specified manually as a *poly*, a class containing three or more points as vertices.

Using known camera and surface geometry, these image regions are rectified (or "un-perspectivized") and registered into common rectangular surface coordinates. Where multiple images contribute intensity values to the same surface region, pixel intensities may not match precisely due to several different factors (which we will explore in section 3.5.1).

One way to sidestep all the problems associated with view-dependent texture differences is to blend textures from different views in the rendering stage as suggested in [90]. In an effort to keep the model simple and compact, we choose instead to attempt to improve texture registration from multiple views and build a static description for texture.

Proposed method for solving local surface deviations

Assuming perfectly diffuse surface reflectivity, the greatest source of view-dependant texture variation is often that the coplanarity assumption is only an approximation. If there are local deviations from planarity and if sufficient texture detail exists we can actually correct for it. As described in [90, 33] given known or pre-calculated cameras, image texture from one subject view can be planar perspective transformed to match the surface as seen in some reference view[12]. Disparities along epipolar lines[13] can be found by intensity normalized correlation.

[11] since scene direction vectors are considered fixed

[12] we choose the one with the most straight on view

[13] An epipolar line for a given point in the reference image is the intersection between the image plane and the epipolar plane for that point which contains that point and the COPs of the reference and subject cameras.

Epipolar disparities are converted to surface disparities by triangulation from the two cameras. Local intensity gradient magnitude can be used as a measure of certainty for surface disparity and relaxation techniques[47] can be used to smooth and fill in areas with low certainty and to force hinge edges at surface boundaries to remain fixed so they properly join with other connected surfaces. This approach can be repeated for all subject views where portions of the surface are visible. Known surface disparities are then used to set height at u-v surface coordinates on a rectangular mesh-grid for each surface.

Proposed method for multi-view texture merging

Given known surface disparities, u-v texture coordinates must be mapped to corresponding x-y image pixels to determine that texture element's color. One way to do this is to temporarily treat each texture element's color as a 6-vector containing its surface I.D., u-v coordinates and 3 terms of the local Jacobian of the projection equation. After normal depth-buffered texture rendering to a 6-channel image, the 6-channels at each x-y pixel describe the surface, u-v surface coordinates and footprint that it projects onto in the scene.

This provides enough information to essentially paint visible surface regions with image pixel colors. From each x-y pixel a color patch is alpha-composited into its u-v surface coordinate with Gaussian extent defined by the Jacobian. This provides a simple framework for correctly merging image intensities from multiple views in a way that uses alpha-blending to preserve available image resolutions.

Before actually using image color values, we may wish to do gamma intensity registration among images to correct for possible variations in exposure. Also, to handle the case of vignetting[14], we give all images alpha transparency as an increasing function of radial distance from the principal point.

Proposed method for computing background plenoptic function

Distant portions of the visible scene are poorly modeled as hinge-connected polyhedral surfaces (the sky in outdoor scenes, for example). Provided all camera orientations have been found, a weighted least-squares estimate of the plenoptic function (discussed in section 2.4.2) from all cameras is a much better model. We use the same framework presented above for normal surfaces, with the exception that camera positions are ignored when projecting u-v plenoptic surface coordinates to x-y image coordinates. The resulting plenoptic function can be used as a special static spherical background image which models infinitely distant objects like horizon and sky.

3.3 Work completed : Two vision-assisted scene modelers

3.3.1 `sceneBuild`

The author has written a bare-bones graphical user interface called `sceneBuild` which uses a `GNU-ReadLine` [108] shell to receive and interpret textual commands. This interpreter client sends messages to a server program which maintains a scene element list as well as several X11 viewer windows which are used to pan/scroll/zoom 2-D images, to create/select/delete/move

[14] Vignetting is a common artifact in real imaging situations where apparent image intensity falls off due to the combined effects of aperture foreshortening and changing aperture distance as seen from a point on the film.

2-D point, line and polygon and to group them into 3-D feature, edge and surface groups. The `sceneBuild` application also supports interprocess message passing commands, so any graphical user interface client can be written, using `Tcl/Tk` [109] for example, to replace or augment texture commands sent by the text-based client. On-line documentation for the `sceneBuild` application may be found at [112] courtesy Bill Butera, `sceneBuild`'s first true user.

3.3.2 `modelMaker`

Building upon `sceneBuild` basic set of scene elements, Eugene Lin has written a more sophisticated modeler called `modelMaker` which uses a `Motif`[15] and `Open Inventor`[16] based graphical user interface.

Eugene, has contributed much to the problem of vision-assisted modeling by developing new user-interface methods for specifying 3-D relationships among 2-D objects. While `sceneBuild` currently only focusses on the use of CAD-like constraints to solve the shape-from-wireframe problem, Eugene's current effort focusses on recovering shape and texture of generalized cylinders from sillouettes in pre-calibrated cameras with pre-estimated pose.

On-line documentation for the `modelMaker` application may be found at [113] courtesy Eugene Lin.

3.4 A proposed application : Model-based video coding

Recent results for affine optical flow segmentation approaches[68, 69, 70] for video coding are promising, but limited success appears to have been made in handling the true perspective case[46, 91]. However, inspiration from Holtzman's ground breaking work in dimensionalization [74], and consideration of recent advancements in self-calibration, structure-from-motion and pose-determination in known environments have prompted a reevaluation of the model-based video coding problem.

3.4.1 Pose-determination in a known scene

We can envision a model-based video coder that is given a previously analyzed set of textured polygons describing a scene. Given new video footage of that scene, a pose-determination stage can be designed which attempts to find the 7 pose parameters that minimize the difference between the synthetic texture rendering of the model and each frame of the observed image.

Section 2.3 above, discusses existing approaches for determining a camera's pose given a known 3-D model and an image of that scene. There we considered that a combination of feature- and iconic-based matching schemes might allow an efficient multi-scale approach. If not given rough a-priori pose, we will use combinatoric search methods for line-based pose determination since these are considered to be most robust in the presence of clutter. Taking the best match I will then do fine scale iconic pose search using summed normalized mean-square error while constraining physical dynamics of a moving camera with non-zero mass. This will hopefully maximize the fit between model and observation at the pixel level while at the same time minimize violations of natural physics for camera motion.

[15] Motif is a standard user interface for UNIX platforms

[16] OpenInventor is an object-oriented toolkit for developing interactive, 3D graphics applications

3.4.2 Actor "sprite" segmentation and playback issues

Having optimally determined the pose for each frame in a batch processing stage, the information which is now transmitted to the decoder consists of the original set of textured polygons, 7 pose parameters for each frame and a (hopefully low bandwidth) residual image signal which compensates for errors in the reproduction.

Of course, typical video sequences have non-rigid elements, like human figures, for which a description involving only a static scene viewed with moving camera is clearly not sufficient. Rather than attempting to recover parameterizations describing human motion or expression, this thesis plans to attack only the human segmentation problem. Proper pose estimation allows us to synthesize the static scene as viewed without unmodeled objects and lighting effects.

Once an uncluttered background image is reconstructed, color histogram approaches like those presented in [77, 86] might allow segmentation of actors and unmodeled objects from each frame.

Failing this, I plan to extend the case of segmenting moving figures in front of a fixed camera in the presence of pixel noise. Rather than using pixel mean and variance of a static background image as the ML[17] estimator at each pixel I plan to compare mean and variance of pixel neighborhoods. This becomes essentially a crude attempt at comparing local texture and color at pixels in the rendered and observed images.

During playback, the Isis (or VRML) decoder has the option of treating these actor sprites as 3-D "billboards" which are always oriented to be parallel to both the scene's vertical axis and the viewer's horizontal axis. Based on the assumption that dynamic objects are always on top of some static object, sprites will be given a 3-D position such that the lowest point in the sprite marks its visible intersection with some horizontal surface (e.g. the ground plane). In this way, the video representation supports either normal 2-D decoding or decoding for interactive 3-D playback in an elegant fashion.

3.4.3 Residual error signal

For the 2-D decoder, residual pixel errors, due to errors in surface modeling, camera pose, actor segmentation, cast shadows, lighting, shading, specularity and translucency will be described in a separate compensating signal. In the best case, this error term will be minimal so that the video representation could be presentable without it. Due to the difficulty in distinguishing among sources of error, no effort will be made to use the pixel error signal directly to adjust the scene model or camera pose.

3.5 Interpretation of errors

One important problem that needs to be investigated is how do we judge the success of the work.

In digital video coding, this judgement is made by weighting lossy coding errors according to visual importance. The human *spatiotemporal visual response function* (or SVRF) describes human responsiveness to different sizes, directions, and velocities of image texture through time. To measure the observability of coder error, we can measure the energy of the error signal after

[17] Maximum Likelihood

filtering by the SVRF. Other less computationally demanding analytical error measures like normalized mean square error are also used; however, all analytical image quality measures are only approximations to and are not successful against real subjective evalutions[92].

Indeed, researchers [98, 99] have explored the limits of the human visual system's with regard to detectability of changes in images over time. Given a set of images of an altered scene even substantial structural changes are undetectable by the human visual system if the eye's difference mechanism can be suppressed. These psychovisual experiments have shown that significant structure and event content changes can be made to static images without the observer being able to detect these changes[18]. However, when the image difference mechanism is in place, the rapid spatiotemporal discontinuities are easily noticed and the attention is drawn there.

3.5.1 A taxonomy of errors for vision assisted modeling

This suggests that topological correctness is more important than real-world geometric truth when modeling static scenes. If an animated sequence made from a topologically correct and realisticallly textured synthetic model does not violate physics of the real world or offend human visualization expectations (e.g. like seeing cracks between surfaces), then the viewer would most likely be unable to point out the difference between the synthetic and real video sequences of a perfectly static scene.

Just so we can see what we're up against, here we list the source of errors that arise when we attempt to recover surface texture and structure from uncalibrated images:

I. Shape failure. The assumptions we make about geometric relationships among features and textures are in fact false.

- points or lines are erroneously matched
- points, lines or textures are not strictly planar causing self occlusion and disparity
- lines are not truely parallel
- angles between lines not perpendicular
- distances between points are incorrect
- lens model does not adequately model true distortion regardless of the number of parameters

II. Sensor limits. There is a limit to accuracy in parameter estimation due to sensor limitations.

- finite limit on image resolution

[18] Experiments show that gross structural changes can be made to a scene such as changing the height of a fence, changing the position of a desk, or replacing grapes with oranges without the observer noticing the difference if the difference mechanism is suppressed

- partial pixel effect
- noisy feature detection/specification

III. Unmodeled elements and effects. This describes errors which remain if structural assumptions and measurements of modeled elements are correct.

- lens effects (glare, lens flare, autogain, vignetting)
- environmental lighting
- view varying texture (non-diffuse effects like specularity and transparency)
- things (chair, person)
- limits in detail: i.e. every single surface can't be modeled
- limits in shape: i.e. every shape can't be modeled

IV. Pixel noise. Even if the synthetic scene is exact, we will see differences between the synthetic and real image.

- imaging effects (glare, autogain, vignetting)
- noise in pixel color

The quality of our estimate will be determined analytically applying the modeling approach to synthetic textured scenes. An error analysis for geometry and texture recovery accuracy will be reported in the final thesis.

3.5.2 Considering errors for model-based video coding

While we did previously point out that analytical measurements for error fail to measure real subjective error; analytical pixel-wise error measures are used by necessity to automatically determine "optimal fit" (e.g. by using weighted least-squares) initially during modeling to find structure, and then during video coding to find camera pose.

Considering the problem of handling error for model-based video coding, the real challenges are: first in finding the best camera pose while ignoring the clutter introduced by figures; and second segmenting figure extent given only the least-squares model as viewed from the least-squares pose. Individual subjective testing of the final reconstructed result will be the best mechanism for judging success or failure of the approach.

3.6 Thesis Deliverables

The intent of this thesis is to develop new algorithms that allow accurate recovery of camera parameters and 3-D structure directly from imagery of man-made scenes. Algorithms will also be developed which exploit prior knowledge of static 3-D scene structure and approximate initial camera position to determine camera motion and detect new unmodeled structures (i.e. actors) in real video sequences taken in the previously modeled scene.

Proof of successful thesis completion will be given using live or prerecorded video demonstration of results obtained from software tools which implement the following new algorithms:

- single view camera calibration and lens distortion correction from vanishing points of straight lines
- structure and motion determination from point and line correspondences with geometric constraints
- robust pose-determination in a known textured scene
- model-based video coding of live actors in a known static scene with a moving camera

4 Schedule and resources

The following is a proposed timeline listing completion dates for different tasks.

February 1996

- find thesis committee.
- submit proposal to DCGS.

March 1996

- finish and test iterative refinement of structure and motion from point and line features with geometric constraints.
- implement surface disparity estimation by model-based stereo.
- demo results of refined scene modeling.
- implement plenoptic function estimation.
- demo results of plenoptic modeling of infinitely distant objects or static scenes under pure rotation.

April 1996

- implement and test pose refinement given coarse pose using feature matching
- demo results of static scene model-based video coding.

May 1996

- implement and test actor segmentation.
- demo results of actor segmentation.

June 1996

- write thesis.

July 1996

- finish thesis.
- defend thesis.

The following is a list of hardware/software resources which will be used for this thesis work. All of these items are presently available and in working order and must remain so in order for the thesis to be completed according to the scheduled timeline.

Cheops stream processor system for video capture, processing

- Intel 80960 cpu.
- m1 board, 1 Gigabyte VRAM board.
- i1 board, 27 Mbytes/sec (640x480x3x30 bytes/sec) video capture board.
- assorted special purpose DSP chips.
- M7 multi-processor operating system.
- Norman real-time dynamic resource allocation server.
- Cheops-host interprocess communications software.
- gcc960 cross platform compiler for Intel 80960 cpu.
- panasonic 3-color CCD video camera or standard video camera.
- video cassette player.

Cheops structured video decoder

- ISIS : structured video playout system.

Cameras and digitization hardware/software

- QuickTake 100 Apple/Kodak or Kodak Digital Camera 40 low resolution digital still cameras
- Minolta 35mm film camera high resolution film camera
- Kodak Photo-CD film negative scanner
- PowerMac with Photo-CD readable drive
- QuickTake software, Photoshop, PBM utilities and ppm2dat Photo-CD to datfile image conversion software

Alpha 300 workstation for analysis, software implementation, and testing

- 24 bit ZLX frame buffer
- Motif Power Toolkit, Motif, X Toolkit, X11 libraries
- Tcl/Tk C-callable interpreter library
- C, C++ compilers

References

[1] R.Y. Tsai, An Efficient and Accurate Camera Calibration Technique for 3-D Machine Vision, *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami Beach, Florida, pp. 364-374, June 22-26, 1986.

[2] R.K. Lenz and R.Y. Tsai, Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3-D Machine Vision Metrology, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 5, September 1988.

[3] D. Zhang, Y. Nomura, and S. Fujii, A simple and accurate camera calibration method, *Proceedings of the SPIE*, Vol. 1822, pp. 139-148, 1992.

[4] Z.C. Lai, On the sensitivity of camera calibration, *Proceedings of the SPIE*, Vol. 1822, 1992.

[5] Y. Nomura, M. Sagara, Hiroshi Naruse, and Atsushi Ide, Simple Calibration Algorithm for High-Distortion-Lens Camera, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 11, pp. 1095-1099, November 1992.

[6] S. Shah and J.K. Aggarwal, A Simple Calibration Procedure for Fish-Eye (High Distortion) Lens Camera, *Proceedings 1994 IEEE International Conference on Robotics and Automation* San Diego, California, pp. 3422-3427, May 8-13, 1994.

[7] S.W. Shih, Y.P. Hung, and W.S. Lin, When should we consider lens distortion in camera calibration, *Pattern Recognition*, Vol. 28. No. 3, pp. 447-461, 1995.

[8] G.P. Stein, Internal Camera Calibration using Rotation and Geometric Shapes, *Thesis for Master of Science in EECS at MIT*, February 1993.

[9] O.D. Faugeras, What can be seen in three dimensions with an uncalibrated stereo rig? *Proceedings Computer Vision EECV'92 Second European Conference on Computer Vision*, p. xv+909, 563-78, Santa Margherita Ligure, Italy, May 18-23, 1992.

[10] O.D. Faugeras, Stratification of three-dimensional vision: projective, affine, and metric representations, *Journal Optical Society of America*, Vol 12, No. 3, pp. 465-484, March 1995.

[11] A. Shashua and N. Navab, Relative affine structure: theory and application to 3D reconstruction from perspective views, In *Proceedings IEEE Conference on CVPR*, Seattle, Washington, (1994).

[12] B. Caprile and V. Torre, Using vanishing points for camera calibration, *IJCV*, Vol. 4, pp. 127-140, 1990.

[13] P. Parodi and G. Piccioli, "3D Shape Reconstruction by Using Vanishing Points", *IEEE Trans. on Pattern Anal. and Mach. Intel.*, Vol. 18, No. 2, pp. 211-217, February 1996.

[14] L.L. Wang and W.H. Tsai, Computing Camera Parameters using Vanishing-Line Information from a Rectangular Parallelipiped, *Machine Vision and Applications*, **3**:129-141, 1990.

[15] Y. Lui and T.S. Huang, Motion estimation from corner correspondences, *Proceedings International Conference on Image Processing*, Singapore, September 5-8, pp. 785-790, 1989.NEED

[16] S.C. Becker and V.M. Bove Jr., Semiautomatic 3-D model extraction from uncalibrated 2-D views, *Proceedings SPIE Visual Data Exploration and Analysis II*, Vol. 2410, pp. 447-461, San Jose, California, February 8-10, 1995.

[17] D.H. Ballard, Generalizing the Hough Transform to Detect Arbitrary Shapes, 1980.CHECK

[18] R.T. Collins and R.S. Weiss, Vanishing Point Calculation as a Statistical Inference on the Unit Sphere, *Machine Vision and Applications*, **3**:400-403, 1990.

[19] E. Lutton, H. Maitre, and J. Lopez-Krahe, Contribution to the Determination of Vanishing Points Using Hough Transform, *IEEE PAMI*, Vol. 16, No. 4. April, 1994, pp. 430-438.

[20] K. Kanatani, Statistical foundation for hypothesis testing of image data, *CVGIP: Image Understanding*, Vol. 60, No. 3, November, pp. 382-391, 1994.

[21] K. Kanatani, Statistical analysis of focal-length calibration using vanishing points, *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 6, December, 1992, pp. 767-775.

[22] K. Kanatani, Computational Cross Ratio for Computer Vision, *CVGIP: Image Understanding*, Vol. 60, No. 3, November, pp. 371-381, 1994.

[23] D. C. Brown, Close-range camera calibration, Presented at the *Symposium on close-range photogrammetry*, Urbana, Illinois, January, 1971.

[24] A. Azarbayejani and A. Pentland, Recursive estimation of motion, structure, and focal length, *IEEE Pattern Analysis and Machine Intelligence*, April, 1994.

[25] B. Horowitz, A. Azarbyejani, T. Galyean, and A. Pentland, "Models from Video", *MIT Media Laboratory Vision and Modeling Technical Report #207*

[26] A. Azarbayejani and A. Pentland, Camera self-calibration from one point correspondence, *MIT Media Laboratory, Perceptual Computing Technical Report #341 Submitted to the IEEE Symposium on Computer Vision*, April, 1995.

[27] R. Szeliski and S.B. Kang, "Recovering 3d shape and motion from image streams using non-linear least squares", *CRL 93/3, Digital Equipment Corporation*, Cambridge Research Labs, Cambridge, MA, March 1993.

[28] T.S. Huang and A.N. Netravali, Motion and Structure from Feature Correspondences: A Review, *Proceedings of the IEEE*, Vol. 82, No. 2, February, pp. 252-268, 1994.

[29] H.C. Longuet-Higgens, A computer algorithm for reconstructing a scene from two projections, *Nature*, **293**(10), pp. 133-135, 1981.

[30] A. Kara, K.M. Wilkes and K. Kawamura, 3D Structure Reconstruction from Point Correspondences between Two Perspective Projections, *CVGIP: Image Understanding*, Vol. 60, No. 3, November, pp. 392-397, 1994.

[31] S. Ullman, *The Interpretation of Visual Motion*, The MIT Press, Cambridge, Ma, 1979.

[32] Carlo Tomasi and Takeo Kanade, Shape and motion from image streams: a factorization method; full report on the orthographic case, *International Journal of Computer Vision*, 9(2):127-154, November, 1992.

[33] Larry Matthies and Takeo Kanade, "Kalman Filter-based Algorithms for Estimating Depth from Image Sequences", *International Journal of Computer Vision*, Vol. 3, pp. 209-236, 1989.

[34] N. Navab, O.D. Faugeras and T. Vieville, The Critical Sets of Lines for Camera Displacement Estimation: A Mixed Euclidean-Projective and Constructive Approach, *ICCV*, 1993.

[35] A. Pentland, A. and S. Sclaroff, Closed-Form Solutions For Physically Based Shape Modeling and Recognition, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 13, No. 7, pp. 715-730, 1991.

[36] D.J. Heeger and J.R. Bergen, Pyramid-Based Texture Analysis/Synthesis, *Computer Graphics Proceedings Siggraph*, Los Angeles, April, pp. 229-238, 1995.

[37] C.J. Taylor and D.J. Kriegman, Structure and Motion from Line Segments in Multiple Images, *Yale University Technical Report No. 0402b*, January 1994.

[38] A. Gelb, *Applied Optimal Estimation*, MIT Press, Cambridge, 1989.

[39] T.J. Broida, S. Chandrashekhar, and R. Chellappa, Recursive 3-D motion estimation from a monocular image sequence, *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 16, No. 4, July, 1990.

[40] N.Cui, J.J. Weng, and P. Cohen, Recursive-Batch Estimation of Motion and Structure from Monocular Image Sequences, *CVGIP: Image Understanding*, Vol. 59, No. 2, March, pp. 154-170, 1994.

[41] S. Mann and S. C. Becker, "Computation of some projective-chirplet-transform and metaplectic-chirplet-transform subspaces, with applications in image processing", *Proceedings DSP World Symposium*, Boston, Massachusetts, November, 1992.

[42] T. Melen, "Extracting physical camera parameters from the 3 by 3 direction linear transformation matrix", *Optical 3-D Measurement Techniques II: Applications in inspection, quality control and robotics*, Wichman Pub., edited by A. Gruen and H. Kahmen, pp. 355-365, 1993.

[43] R. Y. Tsai and T. S. Huang, Estimating Three-Dimensional Motion Parameters of a Rigid Planar Patch, *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-29, No. 6, December, pp. 1147-1152, 1981.

[44] K. Prazdny, On the Information in Optical Flows, *Computer Vision, Graphics and Image Processing*, **22**, pp. 239-259, 1983.

[45] Barron, Fleet and Beauchemin, Systems and Experiment: Performance of Optical Flow Techniques, *International Journal of Computer Vision*, 12:1, pp. 43-77, 1994.

[46] G. Adiv, Determining Three-Dimensional Motion and Structure from Optical Flow Generated by Several Moving Objects, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-7, No. 4, pp. 384-401, July 1985.

[47] B. K. P. Horn, *Robot Vision*, MIT Press, 1986.

[48] S. Negahdaripour, and B. K. P. Horn, Direct Passive Navigation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9:1, January 1987.

[49] Berthold K. P. Horn and E.J. Weldon Jr., Direct methods for recovering motion, *IJCV*, 2, 51-76 1988.

[50] David Heeger and Allan Jepson, Simple method for computing 3D motion and depth, *Proceedings 3rd IEEE International Conference on Computer Vision*, pp. 96-100, 1990.

[51] A.F. Bobick, Using Stability of Interpretation as Verification for Low Level Processing: An example from ego-motion and optic flow, *IEEE Conference on Computer Vision and Pattern Recognition*, New York, June 1993.

[52] T. Marill, Emulating the Human Interpretation of Line-Drawings as Three-Dimensional Objects, IJCV, 6:2, pp. 147-161, 1991.

[53] Yvan G. Leclerc and Martin A. Fischler, An Optimization-Based Approach to the Interpretation of Single Line Drawings as 3D Wire Frames, *IJCV*, 9:2, 113-136 (1992)

[54] R. Nevatia, M. Zerroug, and F. Ulupinar, Recovery of Three-Dimensional Shape of Curved Objects from a Single Image, /em Handbook of Pattern Recognition and Image Processing: Computer Vision, Academy Press, pp. 101-129, 1994.

[55] M.A. Fischler and R.C. Bolles, Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography, *Communications of the ACM*, Vol. 24, No. 6, pp. 381-395, June 1981.

[56] R.M. Haralick, C.N. Lee, K. Ottenberg, M. Nolle, Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem, *IJCV*, Vol. 13, No. 3, pp. 33-356, 1994.

[57] Y. Liu, T.S. Huang, and O.D. Faugeras, Determination of camera location from 2D to 3D line and point correspondences, *Proceedings CVPR*, Ann Arbor, MI, June, 1988.NEED

[58] R. Kumar and A. R. Hanson, Robust Methods for Estimating Pose and a Sensitivity Analysis, *CVGIP: Image Understanding*, Vol. 60, No. 3, November, pp 313-342, 1994.

[59] J.I. Par, N. Yagi, K. Enami, K. Aizawa, and M. Hatori, Estimation of Camera Parameters from Image Sequence for Model-Based Video Coding, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 4, No. 3. June 1994.

[60] J.D. Foley, A. Van Dam, S.K. Feiner and J.F. Hughes, *Computer Graphics Principles and Practice*, 2nd Ed., Addison-Wesley, New York, 1990.

[61] R. Forchheimer and O. Fahlander, Low bit-rate coding through animation, in *Proceedings Picture Coding Symposium (PCS-83)*, Davis, pp. 113-114, March 1983.NEED

[62] H. Li, P. Roivainen and R. Forchheimer, 3-D Motion Estimation in Model-Based Facial Image Coding, *IEEE Transactions on PAMI*, Vol. 15, No. 6, pp. 545-555, June 1993.

[63] I.A. Essa, Analysis, Interpretation and Synthesis of Facial Expression, *MIT Media Laboratory Ph.D. Thesis*, February 1995.

[64] E.H. Adelson, and J.R. Bergen, "The Penoptic Function and the Elements of Early Vision", *Computational Models of Visual Processing*, Chapter 1, Edited by M. Landy and J. A. Mavson, The MIT Press, Cambridge, Mass, 1991.

[65] L. McMillan and G. Bishop, Plenoptic Modeling: An Image-Based Rendering System, Proceedings *ACM Siggraph '95*, pp. 39-46, 1995.

[66] S. Mann and R. Picard, "Video Orbits of the Projective Group: A New Perspective on Image Mosaicing", *MIT Media Laboratory Perceptual Computing Section Technical Report No. 338*.

[67] L. Teodosio, "Salient Stills", *MIT Media Laboratory Masters Thesis*, 1992.

[68] J. Wang, T. Adelson and U. Desai, Applying Mid-level Vision Techniques for Video and Data Compression and Manipulation, *Proceedings of the SPIE: Digital Video and Data Compression on Personal Computers: Algorithms and Technologies*, Vol. 2187, San Jose, February 1994.

[69] M. Irani, S. Hsu, P. Anandan, Mosaic-based video compression, *Proceedings SPIE: Digital Video Compression*, Vol. 2419, pp. 242-253, San Jose, California, February 8-10, 1995.

[70] R.G. Kermode, Building the BIG Picture: Enhanced Resolution from Coding, *MIT Media Laboratory Masters Thesis*, June 1994.

[71] J. Canny, A Computational Approach to Edge Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 6. Nov. (1986), pp. 679-698.

[72] Stefan Agamanolis, "Line-finder man pages", *MIT Media Lab TVOT man pages*, December, 1995.

[73] Stefan Panayiotis Agamanolis, "High-level scripting environments for interactive multimedia systems", *MIT Media Lab*, Masters Thesis, February 1996.

[74] H. Holtzman, Three-Dimensional Representations of Video using Knowledge-Based Estimation, *MIT Media Laboratory Masters Thesis*, 1991.

[75] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical recipes in C*, Cambridge University Press, Cambridge, UK, 1988.

[76] W. T. Freeman, "Steerable Filters and Local Analysis of Image Structure", Ph.D. Thesis, Massachusetts Institute of Technology, 1992. Also available as Vision and Modeling Technical Report 190, MIT Media Laboratory.

[77] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland, Pfinder: Real-Time Tracking of the Human Body, *MIT Media Laboratory Perceptual Computing Section*, Technical Report No. 353, 1995.

[78] W. E. L. Grimson, D. P. Huttenlocher and T. D. Alter, "Recognizing 3D Objects from 2D Images: An Error Analysis", *MIT AI Laboratory*, A.I. Memo No. 1362, July 10, 1992.

[79] K. Nagao and W. E. Grimson, Object Recognition by Alignment using Invariant Projections of Planar Surfaces, *MIT AI Laboratory*, A.I. Memo No. 1463, February, 1994.

[80] K. Nagao and W. E. Grimson, "Recognizing 3D Objects Using Photometric Invarient", *MIT AI Laboratory*, A.I. Memo No. 1523, February, 1995.

[81] K. Nagao and B. K. P. Horn, "Direct object recognition using no higher than second or third order statistics of the image", *MIT AI Laboratory*, A.I. Memo No. 1526, February, 1995.

[82] P. A. Viola, Alignment by Maximization of Mutual Information, *MIT AI Laboratory*, A.I. Technical Report No. 1548, June, 1995.

[83] J. A. Provine and L. T. Bruton, "3-D Model Based Coding - A Very Low Bit Rate Coding Scheme for Video-conferencing", to appear in *Proc. IEEE Intl. Symp. on Circuits and Systems*, 1996.

[84] S. Ullman, An Approach to Object Recognition: Aligning Pictorial Descriptions, *AI Memo, No. 931*, December 1986.

[85] W. Eric L. Grimson and David Huttenlocher, On the Sensitivity of the Hough Transform for Object Recognition, *MIT A.I. Memo No. 1044*, May 1988.

[86] Tanveer Fathima Syeda-Mahmood, "Data and Model-driven Selection using Color Regions", *CVVV'92*, pp. 115-123, 1992. May 1993.

[87] S. Tanveer F. Mahmood, "Data and Model-driven Selection using Parallel-Line Groups", *MIT A.I. Memo, No. 1399*, May 1993.

[88] W.K. Pratt, O.D. Faugeras and A. Gagalowicz, "Applications of Stochastic Texture Field Models to Image Processing", *Proc. of the IEEE*, Vol. 69, No. 5, May 1981.

[89] K. Popat and R.W. Picard, "Novel cluster-based probability model for texture synthesis, classification, and compression", *MIT Media Lab Perceptual Computing Group Technical Report #234*, November 1993.

[90] P.E. Debevec, C. J. Taylor and J. Malik, "Modeling and Rendering Architecture from Photographs", *Technical Report UCB/CSD-96-893*, January 1996.

[91] S. Ayer, P. Schroeter and J. Bigun, "Segmentation of moving objects by robust motion parameter estimation over multiple frames", *CVVP-94*, Vol. 801, pp. 316-327.

[92] R. E. Grant, A. B. Mahmoodi, O. L. Nelson, "Chapter 11: Image Compression and Transmission", *Imaging Processes and Materials*, pp. 323-349.

[93] Roberts.

[94] Ganapathy.

[95] Guzman and Falk.

[96] Clowes.

[97] Huffman.

[98] .

[99] .

[100] http://www.cs.Berkeley.EDU/ sequin/soda/soda.html

[101] http://www.portola.com:80/TR/background.html

[102] http://vrml.wired.com/concepts/raggett.html

[103] http://quasar.poly.edu/ llin/GISS/sessions/SecII-A.html

[104] http://vision.ucsd.edu/

[105] http://sdsc.edu/SDSC/Partners/vrml/doc.html

[106] http://vrml.wired.com/future/scale.html

[107] http://www.photomodeler.com

[108] http://garden-docs.www.media.mit.edu/garden_docs/gnu/readline/rlman_toc.html

[109] http://cuiwww.unige.ch/eao/www/TclTk.html

[110] http://www.osf.org/motif/Motif20/index.html

[111] http://www.sgi.com/Technology/Inventor/

[112] file:///mas/garden/grad/bill/images/modeling/HowTo.html

[113] file:///mas/garden/urop/elin/Projects/ModelMaker/Documentation/ModelMaker_docs.html

[114] http://cheops.www.media.mit.edu/projects/cheops/

[115] http://www.microsoft.com/intdev/avr/avwhite.htm

[116] http://webspace.sgi.com/moving-worlds