DOI: 10.1111/j.1467-8659.2008.01181.x

COMPUTER GRAPHICS *forum*
Volume 27 (2008), number 8 pp. 2197–2218

Camera Control in Computer Graphics

Marc Christie[1], Patrick Olivier[2] and Jean-Marie Normand[3]

[1]IRISA/INRIA Rennes Bretagne Atlantique, Campus de Beaulieu, 35042, Rennes Cedex, France
[2]Culture Lab, School of Computing Science, Newcastle University, United Kingdom
[3]LINA Computer Science Laboratory, FRE CNRS 2729, Nantes University, France

Abstract

Recent progress in modelling, animation and rendering means that rich, high fidelity virtual worlds are found in many interactive graphics applications. However, the viewer's experience of a 3D world is dependent on the nature of the virtual cinematography, in particular, the camera position, orientation and motion in relation to the elements of the scene and the action. Camera control encompasses viewpoint computation, motion planning and editing. We present a range of computer graphics applications and draw on insights from cinematographic practice in identifying their different requirements with regard to camera control. The nature of the camera control problem varies depending on these requirements, which range from augmented manual control (semi-automatic) in interactive applications, to fully automated approaches. We review the full range of solution techniques from constraint-based to optimization-based approaches, and conclude with an examination of occlusion management and expressiveness in the context of declarative approaches to camera control.

Keywords: virtual camera control, camera planning, virtual cinematography

ACM CCS: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism, I.3.6 [Computer Graphics]: Methodology and Techniques

1. Introduction

In an attempt to transpose both photographic and cinematographic techniques to computer graphics environments, a number of approaches have been proposed that enable both interactive and automated control of virtual cameras. Camera control, which encompasses viewpoint computation, motion planning and editing, is a component of a large range of applications, including data visualization, virtual walk-throughs, virtual storytelling and 3D games. Formulating generic camera control techniques, and addressing the intrinsic complexity of computing camera paths (which can be solving with a polynomial complexity) is an on-going challenge.

From an applications perspective, approaches to camera control can be distinguished on the basis of whether the user exercises some degree of interactive control, or the application assumes full control of the camera itself. Interactive approaches propose a set of mappings between the dimensions of the user input device (e.g. mouse, keyboard) and the camera parameters. The nature and complexity of the mappings are highly dependent on the targeted application. Low-level approaches rely on reactive techniques borrowed from robotics and sensor planning, where the behaviour of the camera is driven in direct response to visual properties in the current image. Constraint-based and optimization-based approaches reflect a move towards higher-level control in which the user specifies desired image properties for which the camera parameters and paths are computed using general purpose solving mechanisms. The range, nature and specificity of the properties characterize the expressiveness of the approach.

While most approaches have been developed in response to the specific requirements of an application domain, there are many common difficulties including the number of degrees of freedom, the computational complexity related to any path-planning problem, and the evaluation and avoidance of occlusion. Our presentation of the state-of-the-art in camera control progresses from interactive approaches

Published by Blackwell Publishing, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main Street, Malden, MA 02148, USA.

Submitted: June 2007
Revised: February 2008
Accepted: July 2008

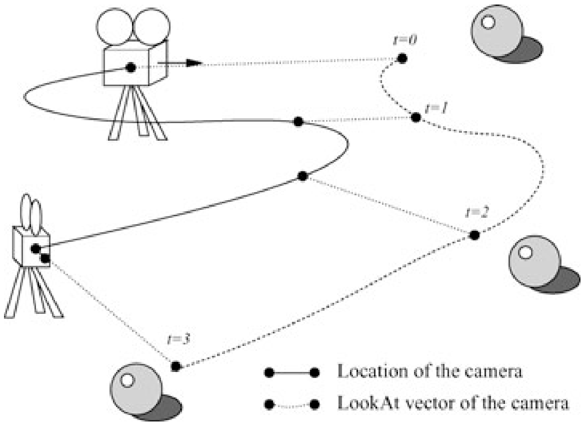


Figure 1: *Canonical specification of a camera path in a 3D modelling environment: two paths define the location and the orientation of the camera (the up-vector is generally represented by the normal vector of the Frenet-Serret frame).*

to fully automated control. After characterizing the requirements of camera control in a number of key applications, we discuss the relevance of photographic and cinematographic practice. We then consider contrasting proposals for user control of a camera and fully automated control. Throughout we emphasize the principal challenges for camera control, and conclude with a discussion of the impact of occlusion and expressiveness on different camera control formulations.

2. Motivations

2.1. Application Context for Camera Control

The requirements for interactive and automated approaches to camera control can in part be found in the use and control of cameras in a number of common computer graphics applications. Though existing applications are bound by the state-of-the art in camera control itself, the goals of the user and existing use of interactive and automated control, provide us useful insights as to the needs of future applications.

2.1.1. *Conventional modellers*

In three-dimensional modeling environments, virtual cameras are typically configured through the specification of the location of the camera and two vectors that represent the *look-at* and *up* directions of the camera. The specification of camera motion is usually undertaken through a combination of direct editing and interpolation, such as the use of splines with key frames and/or control points. As illustrated in Figure 1, animation of the camera is realized by interpolating the camera location, *up* and *look-at* vectors across key-frames. Fine control of camera speed is provided through the ability to manipulate the velocity graphs for each curve.

A set of complementary tools provides modellers with the ability to use the position of a unique static or dynamic target object to constrain the *look-at* vector. Modellers may also allow the use of offset parameters to shift the camera a small amount from the targeted object or path. Similarly, some tools allow constraints to be added to fix each component of the *look-at* vector individually. Physical metaphors are also used to aid tracking, such as virtual rods that link the camera to a target object. With the possibility to extend the functionality of modellers through scripting languages and plugins, new controllers for cameras can be readily implemented (e.g. using physics-based systems). Furthermore, with the rise of image-based rendering, the creation of camera paths using imported sensor data from real cameras is increasingly popular.

In practice, the underlying camera control model (i.e. two spline curves) is not well suited to describing the behavioural characteristics of a real world cameraman, or the mechanical properties of real camera systems. Despite the fact that a number of proposals exist for describing cinematic practice in terms of camera position, orientation and movement (see Section 3), most modellers have not attempted to explicitly incorporate such notions in their tools. Even basic functionality, such as automatically moving to an unoccluded view of a focal object, cannot be found in current commercial modelling environments.

This mismatch can in part be explained by the general utility that most modelling environments strive to achieve. Cinematic terminology is largely derived from character-oriented shot compositions, such as *over-the-shoulder shots*, *close shots* and *mid shots*. Operating in these terms would require the semantic (rather than just geometric) representation of objects. Furthermore, the problem of translating most cinematographic notions into controllers is non-trivial, for example, even the seemingly simple notion of a *shot* will encompass a large set of possible, and often distinct, solutions. However, providing users with high-level tools based on cinematic constructs for the specification of cameras and camera paths, would represent a significant advance over the existing key-frame and velocity graph-based controls.

2.1.2. *Games*

Interactive computer games serve the benchmark application for camera control techniques. Most importantly, they impose the necessity for real-time camera control. A canonical camera control problem involves following one or more characters whilst simultaneously avoiding occlusions in a highly cluttered environment. Furthermore, narrative aspects of real-time games can be supported by judicious choice of shot edits both during and between periods of actual gameplay. The increasing geometric complexity of games means that most deployed camera control algorithms in real-time 3D games rely upon fast (but fundamentally limited) heuristic occlusion checking techniques, such as ray casting (see Section 6 for a full discussion of occlusion).

Camera control in games has received considerably less attention in computer games than visual realism, though as John Giors (a game developer at Pandemic Studios) noted, "the camera is the window through which the player interacts with the simulated world" [Gio04]. Recent console game releases demonstrate an increasing desire to enhance the portrayal of narrative aspects of games and furnish players with a more cinematic experience. This requires the operationalization of the rules and conventions of cinematography. This is particularly relevant in the case of games that are produced as a film spin-offs, where mirroring the choices of the director is an important means of relating the gameplay to the original cinematic experience.

Games are inherently different from film in that the camera is usually either directly or indirectly controlled by players (typically through their control of characters to which a camera is associated). Furthermore, a game is a dynamic and real-time environment and game camera systems must be responsive to action that takes place beyond the focal characters. As emphasized by Halper *et al.* in [HHS01], the enforcement of frame coherency (smooth changes in camera location and orientation) is necessary to avoid disorienting players. While the automation of camera control based on cinematographic principles aims to present meaningful shots, the use of editing techniques (which are rare) can preserve gameplay by presenting jump-cuts or cut-scenes to guide the user. The use of automated editing and cinematographic techniques in games is currently the exception rather than the rule.

Full Spectrum Warrior (cf. [Gio04]), an action war military simulator developed at Pandemic Studios, was a notable progression in the use of cameras in games. The interactive camera control framework successfully supported a player undertaking the complex task of managing teams of soldiers. An important element was the *auto-look* feature, which maintained unoccluded shots of targets through the use of ray casting. Fly-by scenes also used ray casting to avoid collisions with environmental objects, and jump cuts were utilized in situations where the obstacles prevented continuous camera motion.

In general, camera usage in games can be classified as:

- *First person*: users control the camera (giving them a sense of *being* the character in virtual environment). Many games use first person camera views, and the most common genre is the First Person Shooter (FPS), for example, the Doom and Quake series. Camera control is unproblematic, since it is directly mapped to the location and orientation of the character.
- *Third person*: the camera system tracks characters from a distance (generally the view is slightly above and behind the main character) and responds to both local elements of the environment (to avoid occlusion) and the character's interactions (maintaining points of interest in shot). Problems arise when the shot fails to support important events in the game, for example, when a character backs-up against a wall such systems typically default to a frontal view, thereby disrupting the gameplay by effectively hiding the activity of opponents. Furthermore, due to the imprecise nature of the occlusion detection procedures in game camera systems (e.g. ray casting), partial but often significant, occlusion of the main character is a common occurrence.
- *Action replay*: replays are widely used in modern racing or multi-character games where there are significant events that a player might like to review (see Figure 2). It is imperative that these replays are meaningful to the players, such that the elements of the scene and their spatial configuration are readily identifiable.

Figure 2: *In-game screenshot of a Burnout 3 instant replay.*

Interactive storytelling [BST05] presents a number of interesting opportunities for camera control. In particular, the explicit representation of both narrative elements and character roles, relations and emotional states, provides a rich basis on which to select shots and edits automatically.

2.1.3. *Multimodal systems and visualization*

The generation of multimodal output (e.g. natural language and graphics) involves careful coordination of the component modalities. Typically such systems have been developed in the domain of education and training and in particular need to address the problem of coordinating the choice of vantage point from which to display the objects being described, or referred to, linguistically.

For example, a direct linguistic reference to an object (e.g. *the handle on the door*) usually requires that the object (i.e.

the handle) is no more than partially occluded in the shot. To satisfy such coordination constraints, multimodal generation systems have relied heavily on the use of default viewpoints [SF91] from which unoccluded views of the elements of discourse are likely to be achieved. Ray casting is used to trivially accept or reject viewpoints although [BRZL98] address the application of constraint-based camera planning in the development of a prototype intelligent multimedia tutorial system. Alternative approaches use cutaways and ghosting, standard devices in engineering graphics, by which occluding elements of scene are removed either by direct surgery on the polygons, manipulation of the depth buffer [SF93] or object transparency.

Beyond simple object references, the coordination of language and graphics poses a number of interesting problems for camera control. Indeed, such applications are a rich source of constraints on a camera, as the semantics of some spatial terms can only be interpreted by reference to an appropriate perspective. For example, descriptions involving spatial prepositions (e.g. *in front of*, *left of*) and dimensional adjectives (e.g. *big*, *wide*) assume a particular vantage point. For projective prepositions the choice of a deictic or intrinsic reference frame, for example, for the interpretation of *in front*, directly depends on the viewpoint of a hypothetical viewer.

In visualization systems, multidimensional data sets may be mapped to different three-dimensional spatial entities with a view to furnishing users with an intuitive and interactive framework to explore the underlying relations. Typically, such data sets, and the resulting visualizations, are often vast landscapes of geometry within which manual interactive control is extremely difficult. Visualization is an application for which the user requires interactive control to explore and pursue hypotheses concerning the data. However, user interaction in such applications is usually restricted to a small number of navigational idioms, for example, the identification of a number of *interesting* points or regions in the data, and the exploration of the remaining data in relation to these. Automatic camera control and assisted direct camera control, has the potential to greatly enhance interaction with large data sets [DZ94, DH02, SGLM03].

In practice, even partially automated three-dimensional multimedia generation requires an interpretation and synthesis framework by which both the visuospatial properties of a viewpoint can be computed (i.e. the interpretive framework) and the viewpoint controlled according to the constraints arising from the semantics of the language used (i.e. the synthesis framework). Likewise, future scientific and information visualization systems will benefit greatly from intelligent camera control algorithms that are sensitive to both the underlying characteristics of the domain and the task that the user is engaged in. Such adaptive behaviour presupposes the ability to evaluate the perceptual characteristics of a viewpoint on a scene and the capability to modify it in a manner that is beneficial to the user.

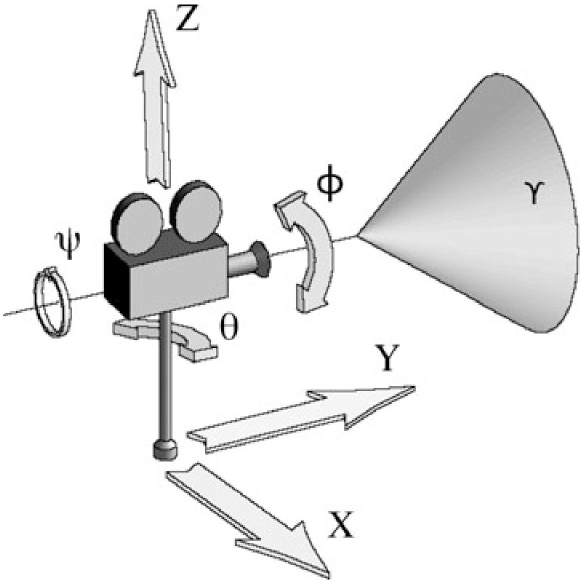


Figure 3: *A simple camera model based on Euler angles; tilt (ϕ), pan (θ) and roll (ψ).*

2.2. Key Issues in Camera Control

Our review of applications incorporating camera control gives rise to a number of fundamental issues. Firstly, direct interactive control of a virtual camera requires dexterity on the part of the user; indeed users find it problematic to deal simultaneously with all seven degrees of freedom. Cameras in computer graphics are modelled using extrinsic parameters, three degrees of freedom for Cartesian coordinates, three Euler angles, and one intrinsic parameter, the *field of view* (see Figure 3). The design of manual control schemes must provide mappings, mediated by interaction metaphors, that meaningfully link the user's actions and the camera parameters.

A second issue is the intrinsic complexity of fully, or partially, automated camera planning. Motion control of virtual camera can be considered as a special case of path planning and is thus a PSPACE-hard problem with a complexity that is exponential in the number of degrees of freedom. Furthermore, the mathematical relation between an object in the 3D scene and its projection on the 2D screen is strongly non-linear. If we consider a Euler-based camera model (see Figure 3) for which the parameters are $\mathbf{q} = [x_c, y_c, z_c, \phi_c, \theta_c, \psi_c, \gamma_c]^T$, then the projection is given by Equation (1). This relation is expressed as a change from the world basis to the local camera basis, with a rotation matrix $R(\phi_c, \theta_c, \psi_c)$, a translation matrix $T(x_c, y_c, z_c)$, and a projection through matrix $P(\gamma_c)$.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = P(\gamma_c) \,.\, T(x_c, y_c, z_c) \,.\, R(\phi_c, \theta_c, \psi_c) \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix} = H(\mathbf{q}) \,.\, \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix} . \qquad (1)$$

The strong non-linearity of this relation makes it difficult to invert (i.e. to decide where to position the camera given both the world and screen locations of the object). Moreover, one must be able to reason about whether the object we are looking at is occluded, either partially or completely, by any other object.

Finally, we need to be able to specify more than simple object layouts on a screen. Camera control assumes the ability to expressively specify desired image properties, this includes perceptual qualities that are meaningful in terms of our visual and spatial cognition; extending beyond the purely geometric. Automated cameras should be capable of assisting users in their construction of mental models and understanding of virtual worlds, conveying the spatial, temporal and causal configuration of objects and events.

3. Camera Control in Cinematography

Direct insight into the use of real-world cameras can be found in accounts of photography and cinematography practice [Ari76, Mas65, Kat91]. Cinematography encompasses a number of issues in addition to camera placement including shot composition, lighting design and staging (the positioning of actors and scene elements), and an understanding of the requirements of the editor. For fictional film, and studio photography, camera placement, lighting design and staging are highly interdependent. However, documentary cinematographers and photographers have little or no control over staging and we review accounts of camera placement in cinematography with this in mind. Indeed, real-time camera control in computer graphics applications (e.g. computer games) is analogous to documentary cinematography whereby coherent visual presentations of the state and behaviour of scene elements must be presented to a viewer without direct modification of the elements themselves.

3.1. Camera Positioning

Whilst characterizations of cinematography practice demonstrate considerable consensus as to the nature of best practice, there is significant variation in its articulation. On the one hand, accounts such as Arijon's systematically classify components of a scene (e.g. according to the number of principal actors) and enumerate appropriate camera positions and shot constraints [Ari76]. Not surprisingly, Arijon's procedural account of camera placement is apparent in the specification of a number of existing automatic camera planning systems (e.g. [Dru94, CAH*96, HCS96, CLCM02, CLDM03, LX05]). Alternative accounts provide less prescriptive characterizations in terms of broader motivating principles, such as narrative, spatial and temporal continuity [Mas65].

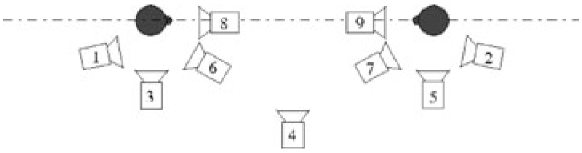


Figure 4: *Arijon's idiom for a two person face-to-face conversation includes nine camera placements in each side of the line of action (top view) [Ari76].*

It is generally considered that camera positioning for dialogue scenes can be characterized in terms of broadly applicable heuristics. For example, Arijon's triangle principle invokes the notion of a line of action, which for single actors is determined by the direction of the actor's view, and for two actors is the line between their heads. Camera positions are selected from a range of standardized shots such as internal-reverse (cameras 6 and 7 in Figure 4), external-reverse (cameras 1 and 2), perpendicular (cameras 3, 4 and 5) and parallel configurations (cameras 8 and 9). By ensuring that camera placements are chosen on one side of the line of action, it can be assured that viewers will not be confused by changes in the relative positions, or the direction of gaze, of the actors caused by camera cuts from one shot to another. In fact, there are a wide range of two actor configurations that vary in respect of the actors' relative horizontal positions (e.g. close together, far apart), orientations (e.g. parallel, perpendicular), gaze direction (e.g. face-to-face, back-to-back) and posture (e.g. sitting, standing, lying down). As a result Arijon enumerates a numerous sets of standard camera positions, and extends the principles for filming two actors to three or more actors in various spatial configurations.

3.2. Shot Composition

Camera positioning determines the relative spatial arrangement of elements of the scene in the shot. That is, the position (and lens selection) determines the class of shot that is achievable. The available shots can be broadly classified according to the amount of the subject included in the shot: close up (e.g. from the shoulders), close shot (e.g. from the waist), medium shot (e.g. from the knee), full shot (e.g. whole body) and long shot (e.g. from a distance). However, precise placement and orientation of the camera determines the layout of the scene elements in shot—referred to as the composition of the shot.

Composition is characterized in terms of shot elements, including lines, forms, masses and motion (in the cases of action scenes). In turn, shots are organized to achieve an appropriate (usually single) centre of attention, appropriate eye scan, unity and compositional balance (i.e. an arrangement of shot elements that affords a subconsciously agreeable picture). As psychological notions these terms are problematic to define and the empirical characterization of visual aesthetics is in its infancy. This is not to question the validity or significance of the notions themselves, indeed eye tracking studies have demonstrated significant differences between the viewing behaviour of people looking at balanced and unbalanced (through artificial modification) works of art [NLK93].

At a practical level, Mascelli observes a number of compositional heuristics, for example, that 'real lines should not divide the picture into equal parts' and that 'neither strong vertical nor horizontal line should be centred' [Mas65]. He further categorizes forms of balance into formal (i.e. symmetrical) and informal (i.e. asymmetrical) balance. Indeed, where scene objects interfere with the composition of a shot, in particular, in close-ups, such objects are frequently removed for the duration of the shot. Note also that elements to be balanced do not necessarily relate to physical objects alone. Important aspects of a composition might include abstract notions such as the line of fire or direction of gaze of an actor. Composition is also constrained by the requirement to produce coherent and continuous cuts between cameras, for example, ensuring that consecutive shots have focii of interest that are roughly collocated.

Algorithms and heuristics for 2D composition have recently been devised for both image editing [KHRO01, LFN04] and static 3D scenes [BDSG04, Bar07, GRMS01]. However, situations in which there are large amounts of motion and action pose significant problems for both virtual and real cinematographers and editors. In such cases general heuristics such as the triangle principle, use of a line of action and compositional rules, can still be applied, but the challenge for automatic camera control is to algorithmically formulate these principles in a manner appropriate to the particular application to be addressed.

4. Interactive Camera Control

4.1. Direct Control

Interactive control systems modify the camera set-up in direct response to user input. The principal design issue is how to map an input device onto the camera parameters. Ware and Osborne [WO90] reviewed the possible mappings or camera control metaphors, and categorized a broad range of approaches:

- *eyeball in hand*: the camera is directly manipulated as if it were in the user's hand, encompassing rotational and translational movement. User input modifies the values of the position and orientation of the camera directly.
- *world in hand*: the camera is fixed and constrained to point at a fixed location in the world, while the world is rotated and translated by the user's input. User input modifies the values of the position and orientation of the world (relative to the camera) directly.
- *flying vehicle*: the camera can be treated as an airplane. User input modifies the rotational and translational velocities directly.
- *walking metaphor*: the camera moves in the environment while maintaining a constant distance (height) from a ground plane [HW97, FPB87].

Where highly interactive control of a spatially localized world is required, the *world in hand* metaphor is highly appropriate. For example, Phillips *et al.* used the *world in hand* metaphor for the human figure modelling system *Jack* [PBG92]. However, despite its intuitive nature, the *Jack* system could not properly support manipulation of model figures about axes parallel to, or perpendicular, to the viewing direction. The camera system prevented this from occurring through a subtle repositioning of the camera. A repositioning of the camera to make a selected object visible (if it was off screen) was also possible. *Jack*'s camera control system could also find positions from which a selected object is unoccluded by placing a non-viewing camera with a fish eye lens at the centre of the target object looking towards the current viewing camera and using the z-buffer to identify clear lines of sight.

For similar applications, Shoemake [Sho92] introduced the concept of the arcball, a virtual ball that contains the object to be manipulated. His solution relies on quaternions to stabilize the computation and avoid Euler singularities (i.e. gimbal lock) while rotating around an object. For a more detailed overview of possible rotation mappings see Chen *et al.*'s study of 3D rotations using 2D input devices [CMS88].

The *flying vehicle* metaphor is widely exploited within computer graphics applications and is generally accepted as an intuitive way to explore large 3D environments such as those that arise in scientific visualization [SS02, BJH01]. In implementing flying vehicle metaphors, a key concern is avoiding the *lost in space* problem that users can encounter when attempting to manage multiple degrees of freedom in either highly cluttered environments, or open spaces with a few visual landmarks. This is typically addressed by reducing the dimensionality of the control problem, and/or the application of physics-based models, vector fields or path planning to constrain possible movement and avoid obstacles [HW97].

For example, the application of a physical model to camera motion control has been explored by Turner *et al.* [TBGT91].

User inputs are treated as forces acting on a weighted mass (the camera) and friction and inertia are incorporated to damp degrees of freedom that are not the user's primary concern. Turner's approach is easily extended to manage any new set of forces and has inspired approaches that rely on vector fields to guide the camera parameters. Given knowledge of an environment this process consists of computing a grid of vectors (forces) that influence the camera. The vectors keep users away from cluttered views and confined spaces, as well as guiding them towards the objects of interest [HW97, XH98].

4.2. Through-the-Lens Control

Whilst direct control metaphors ease the problem of the interactive control of camera parameters, none are concerned with a precise control of the movement of objects in the screen. Gleicher and Witkin's *Through The Lens Camera Control* [GW92] allows the user to control a camera by manipulating the locations of objects directly on the screen. A recomputation of new camera parameters is performed to match the user's desired location. The difference between the actual screen location and the desired location, as indicated by the user, is treated as a velocity and the relationship between the velocity ($\dot{\mathbf{h}}$) of m displaced points on the screen, and the velocity ($\dot{\mathbf{q}}$) of the camera parameters can be expressed through the Jacobian matrix that represents the perspective transformation:

$$\dot{\mathbf{h}} = J\dot{\mathbf{q}}.$$

Gleicher and Witkin present this as a non-linear optimization problem in which a quadratic energy function $E = \frac{1}{2}(\dot{\mathbf{q}} - \dot{\mathbf{q}}_0) \cdot (\dot{\mathbf{q}} - \dot{\mathbf{q}}_0)$ is minimized. This represents a minimal change in the camera parameters (where $\dot{\mathbf{q}}_0$ represents the values of the camera's previous velocity). This problem can be converted into a Lagrange equation and solved for the value of λ:

$$\frac{dE}{d\dot{\mathbf{q}}} = \dot{\mathbf{q}} - \dot{\mathbf{q}}_0 = J^T\lambda,$$

where λ stands for the vector of Lagrange multipliers. The velocity of the camera parameters is thus given by

$$\dot{\mathbf{q}} = \dot{\mathbf{q}}_0 + J^T\lambda.$$

A simple Euler integration allows us to approximate the next location of the camera from the velocity $\dot{\mathbf{q}}$:

$$\mathbf{q}(t + \Delta t) = \mathbf{q}(t) + \Delta t\ \dot{\mathbf{q}}(t).$$

The result is that the rate of change of the camera set-up is proportional to the magnitude of the difference between the actual screen properties and the desired properties set by the user. When the problem is over-constrained (i.e. the number of control points is higher than the number of degrees of freedom) the complexity of the Lagrange process is $O(m^3)$.

This formulation has been improved and extended by Kung, Kim and Hong [KKH95] with the use of a single Jacobian matrix. A pseudo inverse of the matrix is computed with the Singular Value Decomposition (SVD) method for which the complexity is $O(m)$. The SVD method enjoys the property that the pseudo inverse always produces a solution with the minimal norm on the variation of the camera parameters $\mathbf{q}$.

4.3. Assisted Control

Assisted control refers to approaches in which a certain knowledge of the environment is utilized to assist the user in his navigation or exploration task. Such approaches are split according to their local or global awareness of the 3D scene.

4.3.1. *Object-based assistance*

Khan *et al.* [KKS*05] propose an interaction technique for proximal object inspection that automatically avoids collisions with scene objects and local environments. The *hovercam* tries to maintain the camera at both a fixed distance around the object and (relatively) normal to the surface, following a hovercraft metaphor. Thus the camera easily turns around corners and pans along flat surfaces, while avoiding both collisions and occlusions. Specific techniques are devised to manage cavities and sharp turns (see Figure 5). A similar approach has been proposed by Burtnyk *et al.* [BKF*02], in which the camera is constrained to a surface defined around the object to explore (as in [HW97]). The surfaces are designed to constrain the camera to yield *interesting* viewpoints of the object that will guarantee a certain level of quality in the user's exploratory experience, and automated transitions are constructed between the edges of different surfaces in the scene.

4.3.2. *Environment-based assistance*

Environment-based assistance, for which applications are generally dedicated to the exploration of complex environments, requires specific approaches that are related to the more general problem of path planning. Applications can be found both in navigation (searching for a precise target) and in exploration (gathering knowledge in the scene). Motion planning problems in computer graphics have mostly been inspired by robotics utilizing techniques such as potential fields, cell decomposition and roadmaps.

Potential fields originated in theoretical physics and the study of charged particle interactions in electrostatic fields. A path-planning problem can be modelled by considering both the obstacles and the camera as similarly charged particles. The solving process is based on a series of local moves following the steepest descent [Kha86]. A field function F is defined as the sum of attractive potentials (the targets) and

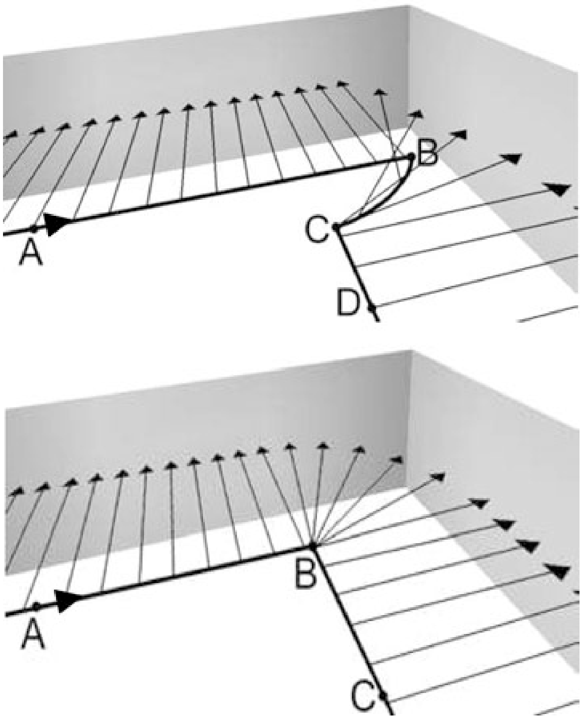


Figure 5: *To avoid collisions with scene elements hovercam [KKS*05] uses a look-ahead in the direction of the camera motion when computing the proximal point.*

repulsive potentials (the obstacles). For efficiency, repulsive potentials attached to obstacles are usually set to zero outside a given distance from the obstacle. Then, the gradient of the field function F, at position p, gives the direction of the next move:

$$M(p) = -\nabla F(p).$$

The low cost of implementation and evaluation of potential fields make them a candidate for applications in real-time contexts.

The efficiency of the method is, however, overshadowed by its limitations with respect to the management of local minima as well as difficulties incorporating highly dynamic environments. Nonetheless, some authors have proposed extensions such as Beckhaus [Bec02] who relies on dynamic potential fields to manage changing environments by discretizing the search space using a uniform rectangular grid and therefore only locally re-computing the potentials.

Cell decomposition approaches split the environment into spatial regions (cells) and build a network that connects the regions. Navigation and exploration tasks utilize this cell connectivity while enforcing other properties on the camera. For example, [AVF04] proposed such a technique to ease the navigation process and achieve shots of important entities and locations. Using a cell-and-portal decomposition of the scene together with an entropy-based measure of the relevance of each cell (see Figure 6), critical way-points for the path could be identified.

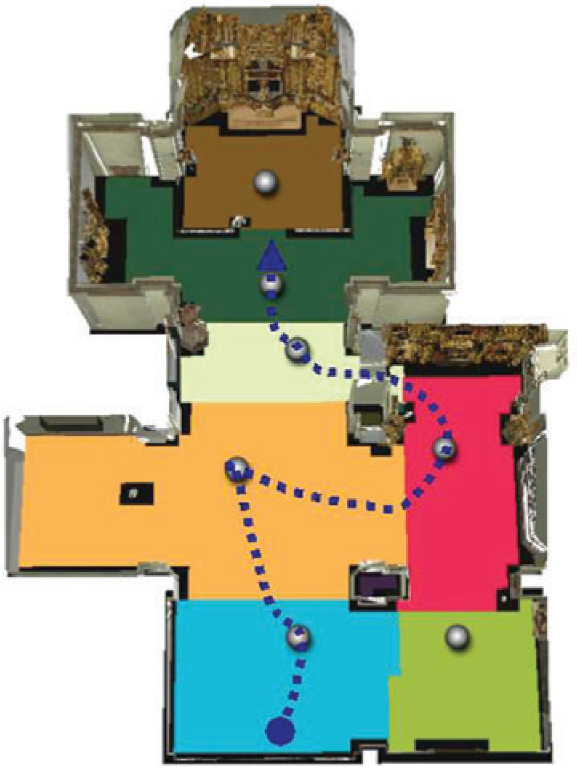


Figure 6: *Cell decomposition and path planning. A cell-and-portal model partitions the search space and the relevance of each cell is evaluated with an entropy-based measure. A path is built by ordering and connecting the most interesting cells (courtesy of Andujar, Vázquez and Fairen, Universitat Politècnica de Catalunya).*

Roadmap planners operate in two phases, by first sampling the space of possible configurations, and then constructing a connectivity graph linking possible consecutive configurations. Probabilistic roadmap approaches, in which the samples are randomly chosen, have been used to compute collision-free paths to correct a user's input, as described in [LT00]. The authors first build a connectivity graph between randomized configurations in the search space. Then, during the interactive navigation process, a set of possible paths are selected according to the current configuration and the user's input, from which the shortest and smoothest one is chosen. The application is restricted to architectural walk-throughs but clearly improves the navigation experience. Salomon *et al.* [SGLM03] describe a related approach in which a global roadmap enables the creation of collision-free and constrained path for an avatar navigating an environment. Nieuwenhuisen and Overmars provide a detailed discussion of the application of robotic techniques to planning camera movements [NO03].

A small but active field for which camera path-planning techniques play an important role is virtual endoscopy. Virtual endoscopy enables the exploration of the internal structures of a patient's anatomy. Difficulties arise in the interactive control of the camera within the complex internal structures. Ideally important anatomical features should be emphasized and significant occlusions and confined spaces be avoided. The underlying techniques mostly rely on skeletonization of the structures and on path-planning approaches such as potential fields. For example, [HMK*97] and [CHL*98] report a technique that avoids collisions for guided navigation in the human colon. The surfaces of the colon and the centre line of the colon are modelled with repulsive and attractive fields, respectively.

Most of the approaches discussed have been developed for tasks that include either close object inspection or large environment exploration. However, a number of techniques have concentrated on transitions between these spatial contexts. Drucker *et al.* [DGZ92] proposed CINEMA, a general system for camera movement. CINEMA was designed to address the problem of combining the different metaphors for direct interactive viewpoint control: *eyeball in hand*; *scene in hand*; and *flying vehicle*. CINEMA also provides a framework in which the user can develop new control metaphors through a procedural interface that allows the specification of camera movements relative to objects, events and general properties of the virtual environment. Zeleznik in [ZF99] demonstrates the utility of this approach by proposing smooth transitions between multiple interaction modes with simple gestures on a single 2-dimensional input device.

Mackinlay *et al.* [MCR90] proposed a means for achieving natural camera transitions between a set of targets that allows each target to be closely inspected. For each transition, this involved a three stage process: (1) view the target; (2) move the camera towards the target at a speed proportional to the target's proximity; and (3) swivel around the object to propose the best view. In a contribution to assisted navigation for large information spaces, Dennis and Healey [DH02] propose a mix between local exploration (by best viewpoint computation) and global exploration by spline interpolation. In contrast to a number of approaches, the authors ensure an interesting background context during the interpolation between best viewpoints.

A helicopter metaphor has been suggested by Jung *et al.* [JPK98] as a mapping function in which transitions are facilitated by a representation of the six degrees of freedom of the camera as a set of planes. Transitions between planes can be readily effected with a 2-dimensional input device.

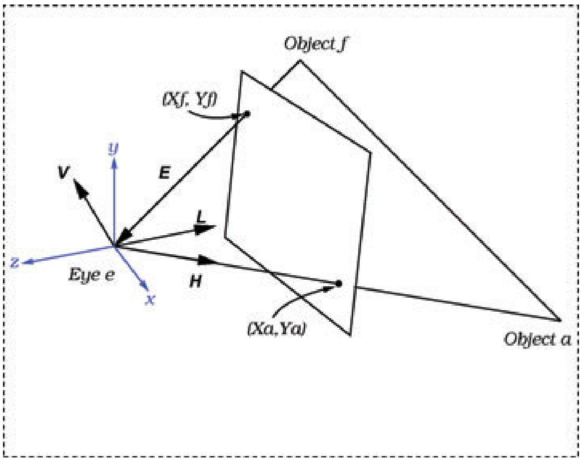


Figure 7: *Blinn's algebraic approach to camera control: two points on the screen, the field of view and an up-vector allow direct computation of the camera position [Bli88].*

More recently, Tan *et al.* in [TRC01] utilized the locations of a user's mouse dragging operations to alternate between the walking around, overview and close object examination interaction metaphors. Li and Hsu [LH04] explored two adaptive methods which personalized an optimal set of control parameters for a user, significantly improving their navigation performance. A contrastive analysis of approaches to direct and indirect camera control are presented by Bowman *et al.* [BJH01] in their taxonomy of interaction techniques and evaluations.

There are a number of possible mappings between user inputs and camera parameters. The progression towards high-level interaction metaphors can be seen as laying the groundwork for the process of automating the placement of a camera. However, fully automatic camera viewpoint control and motion planning control requires the expression of a viewer's goals in terms of shot properties, and the realization of these shots requires the development of both appropriately expressive representation schemes and efficient solution techniques.

5. Automated Camera Control

The earliest example of an automated camera system was Blinn's work at NASA [Bli88]. Whilst working on the visualization of space probes passing planets, he developed a framework for configuring a camera so that the probe and planet appeared on screen at given coordinates. The problem was expressed in terms of vector algebra for which both an iterative approximation and a closed form solution could be found. Blinn's approach requires the world coordinates of the objects f and a (see Figure 7), their desired location on the screen (X_f, Y_f and X_a, Y_a), the *up-vector*, and the aperture of the camera. The closed form solution computes the parameters of the translation matrix T and rotation matrix R of the transformation from world space to view space (see Equation 1). As for most vector algebra approaches, the solution is prone to singularities. Interestingly, the numerical solution has the desirable property of producing approximate results even where the specified properties have no solution.

Solutions typically have the advantage of being efficient to compute, although the necessary idealization of objects as points, and the restriction to locating two entities in the screen space, limits their application. For example, the range of problems that Blinn could specify was restricted to those involving one spaceship and one planet. Attempts to generalize such approaches have relied on the use of film idioms—standard layouts of subjects and cameras commonly used in cinematography. In such systems the solution methods are devised for each layout, and the input consists of a model world together with a list of idioms to apply [But97, CAH*96].

Similar techniques have also been applied to 2D applications such as cell animation (e.g. motion-picture cartoons) and virtual guided tours (e.g. the presentation of sequences of artworks such as frescos, tapestries and paintings). The use of camera control algorithms in cartoon animation was addressed by Wood *et al.* [WFH*97] . Here the problem was to generate a single 2D background image (called a *multiperspective panorama*) from a 3D scene and camera paths. The panoramas were generated by relying on some stock camera moves such as pan, tilt-pan, zoom and truck. Once the *multiperspective panoramas* and their associated *moving windows* have been generated, other computer-animated elements can be incorporated.

Another field of application for 2D camera planning is *multimedia guided tours*, for example, audiovisual guides to artworks at a heritage site. Zancanaro *et al.* explored the use of personal digital assistants in such multimedia museum tours [ZSA03, ZRS03] and their system computed camera movements over *still* images (the frescos on the walls of a room). Both the shot and shot transitions directed the attention of visitors to important aspects of the artworks and small details that could be difficult to identify with an audio commentary alone. Similarly, Palamidese's algebraic approach [Pal96] generates descriptions of a painting by first planning camera movements that show details, and then zooming out to show these details in the context of the entire artwork.

The application of most of these methods is limited by the requirement to represent objects as points. Furthermore, such abstractions will inevitably result in contrived camera configurations where this approximation is inappropriate, for example, where the objects differ significantly in size or have complex shapes. In practice, limitations in the expressive power of vector algebra techniques (on which most of these approaches rely) mean that they are insufficient for most real world camera control problems.

5.1. Reactive Approaches

Reactive approaches involve the computation of a direct response to changes in the image properties without resorting to any search process. In camera control, reactive approaches have been very much inspired by the robotics community. For example, visual servoing approaches (also called image-based camera control) are widely deployed [ECR93]. Visual servoing involves the specification of a task (usually positioning or target tracking) as the regulation of a set of visual features in an image.

In [CM01], a visual servoing approach is proposed that integrates constraints on a camera trajectory in order to address various non-trivial computer animation problems. A Jacobian matrix expresses the interaction between the movement of the object on the screen and the movement of the camera. The solving process involves computing the possible values of the camera velocity in order to satisfy all the properties. If $\mathbf{P}$ is the set of image features used in the visual servoing task; to ensure the convergence of $\mathbf{P}$ to its desired value $\mathbf{P_d}$, we need to know the interaction matrix (the image Jacobian) $\mathbf{L}_{\boldsymbol{P}}^{T}$ that links the motion of the object in the image to the camera motion. The convergence is ensured by [ECR93]:

$$\dot{\mathbf{P}} = \mathbf{L}_{\mathbf{P}}^{T}(\mathbf{P}, \mathbf{p})\mathbf{T_c}, \qquad (2)$$

where $\dot{\mathbf{P}}$ is the time variation of $\mathbf{P}$ (the motion of $\mathbf{P}$ in the image) due to the camera motion $\mathbf{T_c}$. The parameters $\mathbf{P}$ in $\mathbf{L}_{\boldsymbol{P}}^{T}$ are scene features, three-dimensional geometric features rigidly linked to the objects concerned. A vision-based task $\mathbf{e_1}$ is defined by:

$$\mathbf{e_1} = \mathbf{C}(\mathbf{P} - \mathbf{P_d}),$$

where $\mathbf{C}$, the combination matrix, has to be chosen such that $\mathbf{C}\mathbf{L}_{\boldsymbol{P}}^{T}$ is full rank along the trajectory of the tracked object. If $\mathbf{e_1}$ constrains all 6-*dof*s of the camera, it can be defined as $\mathbf{C} = \mathbf{L}_{\mathbf{P}}^{T^+}(\mathbf{P}, \mathbf{p})$, where $\mathbf{L}^+$ is the pseudo inverse of the matrix $\mathbf{L}$. The camera velocity is controlled according to the following relation: $\mathbf{T_c} = -\lambda\mathbf{e_1}$, where λ is a proportional coefficient.

If the primary task (following the object) does not instantiate all the camera parameters when solving Equation (2), secondary tasks may be added (e.g. avoiding obstacles or occlusions, lighting optimization). $\mathbf{C}$ is then defined as $\mathbf{C} = \mathbf{C}\mathbf{L}_{\boldsymbol{P}}^{T}$ and we obtain the following task function:

$$\mathbf{e} = \mathbf{W}^+\mathbf{e_1} + (\mathbf{I_n} - \mathbf{W}^+\mathbf{W})\mathbf{e_2}, \qquad (3)$$

where:

- $\mathbf{e_2}$ is a secondary task. Usually $\mathbf{e_2}$ is defined as the gradient of a cost function h_s to minimize ($\mathbf{e_2} = \frac{\partial h_s}{\partial r}$), which is minimized under the constraint that $\mathbf{e_1}$ is realized.

- $\mathbf{W}^+$ and $\mathbf{I_n} - \mathbf{W}^+\mathbf{W}$ are two projection operators which guarantee that the camera motion due to the secondary task is compatible with the regulation of $\mathbf{P}$ to $\mathbf{P_d}$.

Given a judicious choice of matrix $\mathbf{W}$, the realization of the secondary task will have no effect on the vision-based task, i.e. $\mathbf{L}_P^T(\mathbf{I_n} - \mathbf{W}^+\mathbf{W})\mathbf{e_2} = 0$. This feature ensures that adding secondary tasks cannot affect the results obtained for the primary task, and therefore cannot invalidate solutions. Secondary tasks, as proposed in [MH98, MC02], include tracking another dynamic object, avoiding obstacles or occlusions and cinematographic notions (e.g. panning, travelling, optimizing lighting conditions). Specifying a secondary task requires the definition of a minimization function h_s. For example, obstacle avoidance can be handled by a cost function that will express the inverse of the distance between the camera and the obstacle. A simple cost function for obstacle avoidance is given by:

$$h_s = \alpha \frac{1}{2\|C - O_c\|^2}, \tag{4}$$

where $C(0, 0, 0)$ is the camera location and $O_c(x_c, y_c, z_c)$ are the coordinates of the closest obstacle to the camera (see [MC02] for examples of secondary task cost functions).

Visual servoing approaches are computationally efficient and thus suitable for highly dynamic environments such as computer games. However, one cannot determine in advance which degrees of freedom of the camera will be instantiated by the main task $\mathbf{W}^+\mathbf{e_1}$. Moreover, maintaining smooth paths requires an additional process to dampen sudden modifications of the camera speed and direction in response to the motion of a target.

5.2. Optimization-Based Approaches

Approaches that address camera control with pure optimization techniques express shot properties as objectives to be maximised. Metrics are provided to evaluate the quality of a shot with respect to the underlying graphical model of the scene and the user's description of the problem. Classical optimization techniques encompass deterministic approaches such as gradient-based or Gauss-Seidel methods and non-deterministic approaches such as population-based algorithms (e.g. genetic algorithms), probabilistic methods (Monte Carlo) and stochastic local search methods. The problem can be expressed as the search for a camera configuration $\mathbf{q} \in \mathbf{Q}$ (where $\mathbf{Q}$ is the space of possible camera configurations) that maximises a fitness function as follows:

$$\text{maximise} \quad F(f_1(\mathbf{q}), f_2(\mathbf{q}), \cdots, f_n(\mathbf{q})) \quad \text{s.t.} \quad \mathbf{q} \in \mathbf{Q},$$

where $f_i : \mathbb{R}^7 \to \mathbb{R}$ measures the fitness of each property and $F : \mathbb{R}^7 \to \mathbb{R}$ aggregates the functions f_i. In its simplest representation, F is generally a linear combination of scalar weighted functions:

$$F(f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_n(\mathbf{x})) = \sum_{i=1}^{n} w_i f_i(\mathbf{x})$$

Discrete approaches address the complexity of exploring this 7-*dof* continuous search space by considering a regular discretization on each degree of freedom. In an approach to virtual camera composition that improves over their CONSTRAINTCAM system, Bares *et al.* [BTMB00] propose the use of a complete search of a discretization of the search space. Using a global optimization process, each configuration is provided a value representing its fitness and an exhaustive generate-and-test process is employed. The fitness is expressed as the aggregation of the satisfaction of each property provided by the user. The search space typically is covered by a 50 × 50 × 50 grid of camera locations, every 15° angle for orientation and 10 possible values for the *field of view*.

The efficiency of the computation is improved by narrowing the discretization to feasible regions of the search space, Bares *et al.* [BMBT00]. The feasible regions are built from the intersection of individual regions related to each image property and reducing the grid resolution at each step. The process terminates either when a given quality threshold is reached when evaluating a candidate or when a minimal grid resolution is obtained.

In CAMPLAN, Olivier *et al.* [OHPL99] followed a similar principle in addressing the visual composition problem (i.e. static camera positioning) as a pure optimization process using genetic algorithms. A large set of properties are utilized including explicit spatial relationships between objects, partial and total occlusion, and size. The fitness function is a linear-weighted combination of the fulfilment of the shot properties. The seven parameters of the camera are encoded in the allele; a population of cameras is then randomly distributed in the search space. Each individual of this population is evaluated with respect to the set of objective functions. The top 90% of the population survives to the next generation and selection is by binary tournament. The remaining 10% is re-generated by random crossover and/or mutation. This was extended for a dynamic camera by optimizing the control points of a quadratic spline (with a fixed *look-at* point and known start and end positions) [HO00].

The computational cost, as well as the non-deterministic behaviour, are the main shortcomings of genetic algorithm-based approaches. By modelling the bounds of the properties, the size of the search space can be significantly reduced [Pic02]. Following the same declarative scheme, feasible locations for the camera are abstracted from the specification of the shot. For example, if a user desires to view the front of an object, the volume of space corresponding to rear shots can be pruned. Where multiple objects and properties are concerned, the final space to search is the intersection of all component feasible regions. In [Pic02] these feasible

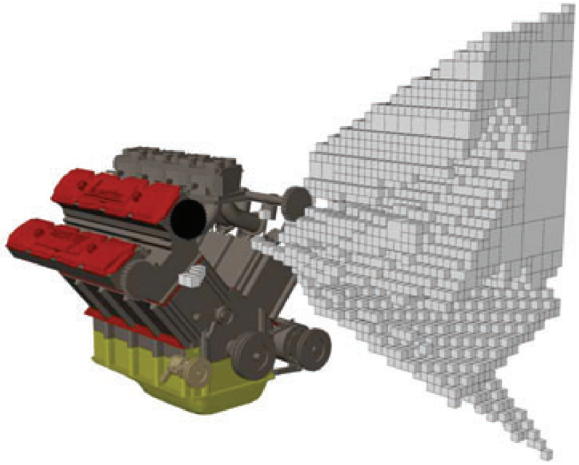


Figure 8: *Intersection of feasible regions modelled using octrees [Pic02].*

regions were modelled in a three-stage process (see Figure 8) as follows:

(i) the construction of a Binary Space Partition (BSP) tree of the shadow volumes using unoccluded objects as light sources (cf. Section 6);
(ii) construction of the feasible regions of other image properties using octrees (e.g. vantage angles and relative position);
(iii) pruning of the octrees such that retained nodes are those that fully lie inside the BSP tree (i.e. satisfy the occlusion constraint).

The structure exploited in the design of the allele comprises a reference to a voxel and an offset inside the voxel (useful for large voxels), an orientation, and a *field of view*. Each allele is subject to crossover operations and the allele design ensures that the search is limited to feasible regions only. As with all optimization approaches, the main problems are the modelling and aggregation of multiple properties into a single objective function.

In an approach to automate camera control in a real-time environment, Halper *et al.* [HHS01] proposed an incremental solving process based on successive improvements of a current configuration, for target-tracking applications. Their camera engine encompasses a large set of shot properties including relative elevation, size, visibility and screen position. Constraints are continuously re-evaluated and in contrast to purely reactive applications of constraints (for example, see [BL99]), Halper *et al.* avoid 'jumpiness' in the camera's movement by maintaining frame-coherence. An algebraic incremental solver computes the new camera configurations from existing camera states. The solving process incrementally satisfies the set of screen constraints at each frame, relaxing subsets as required. Moreover, look-ahead techniques are used to adjust the camera parameters in anticipation of an object's future state. Bourne and Sattar [BS05b] proposed a related optimization-based approach that uses local search techniques to maintain properties both relative to the object of interest (height, distance, orientation) and on the camera path (frame-coherence).

The general problem of computing good viewpoints of 3D environments is shared by a number of applications in both computer graphics and robotics. Image-based modeling, for example, requires the computation of a minimal set of cameras covering all visible surfaces of a 3D environment (for applying captured textures to virtual objects). Most approaches take as inspiration Kamada and Kawai's early contribution [KK88] that aims to find views that maximise the *projected area* to *surface area* ratio. Approaches either resort to classical solvers (Stuerzlinger *et al.* employ simulated annealing [Stu99]) or heuristic search approaches that populate the environment with cameras and provide a covering metric to discriminate solutions (as in Fleishman *et al.* [FCOL00]). The use of viewpoint entropy as a metric has been explored by Vázquez *et al.* [VFSH03] to maximise the quantity of information in a minimal set of views.

In a similar vein a number of researchers have attempted to characterise cognitive aspects such as scene understanding and attention. In Viola *et al.* [VFSG06], the geometry is augmented with object importance, and the optimization process uses visibility and importance in computing a set of characteristic views. Another approach, that focuses more on scene understanding, generates automatic camera paths. At each step, the next viewpoint is computed by a heuristic optimization technique that relies on a local neighbourhood search based on a physical model [SP06] that attracts the camera towards unexplored areas of the scene. The initial configuration is computed by a viewpoint quality estimation algorithm that relies on the computation of total curvature of the visible surfaces together with its projected area.

5.3. Constraint-Based Approaches

Constraint satisfaction problem (CSP) frameworks offer a declarative approach to modelling a large range of mainly non-linear constraints and propose reliable techniques to solve them. Interval arithmetic-based solvers compute the whole set of solutions as interval boxes (a Cartesian product of intervals). Snyder [Sny92] presented a broad review of the application of interval arithmetic in computer graphics. Each unknown in the problem is considered as an interval bounded by two floating points that represent the domain within which the search should be conducted. All the computations therefore integrate operations on intervals rather than on floating point values. The interval extension for the classical operators are as follows:

$[a, b] + [c, d] = [a + c, b + d]$
$[a, b] - [c, d] = [a - d, b - c]$
$[a, b] \times [c, d] = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)]$
if $0 \notin [c, d]$
$[a, b] \div [c, d] = [\min(\frac{a,c], \frac{a}{d}, \frac{b}{c}, \frac{b}{d})}{} \max(\frac{a}{c}, \frac{a}{d}, \frac{b}{c}, \frac{b}{d})$.

In a similar way, all other operations can be extended to interval operators in such a way that for an operator $\top$ the following relation holds:

$$[a, b] \top [c, d] = \{x \top y \mid x \in [a, b] \text{ and } y \in [c, d]\}.$$

The fundamental property of interval arithmetic, on which the solving process relies and correctness is guaranteed, is the containment property [Moo66]:

$$\forall \mathbf{x} \in \mathbb{R} | \mathbf{x} \in \mathbf{X}, f(\mathbf{x}) \subset \mathbf{F}(\mathbf{X}),$$

where X is an interval, $f(\mathbf{x})$ a unary function and $\mathbf{F}(\mathbf{X})$ its interval extension. The interval extension of a function f is a function $\mathbf{F}$, where all the operators on real values have been replaced by operators on interval values. This property states that an interval evaluation of an interval function $\mathbf{F}$ always contains the evaluation of the real function f. For example, the solution set of the constraint $f(\mathbf{x}) \geq 0$ can be approximated by a three-valued logic True, False, Unknown:

$$\begin{cases} \text{if } \mathbf{F}(\mathbf{X}) \cap [0, +\infty] = \oslash, & \text{return False} \\ \text{if } \mathbf{F}(\mathbf{X}) \cap [0, +\infty] = \mathbf{F}(\mathbf{X}), & \text{return True} \\ \text{if } \mathbf{F}(\mathbf{X}) \cap [-\infty, 0] \neq \oslash, & \text{return Unknown} \end{cases}$$

and considering a constraint equation $f(\mathbf{x}) = 0$:

$$\begin{cases} \text{if } \mathbf{F}(\mathbf{X}) \supset 0, & \text{return Unknown} \\ \text{if } \mathbf{F}(\mathbf{X}) \not\supset 0, & \text{return False} \end{cases}.$$

This leads to a dynamic programming evaluate-and-split process as presented in Figure 9. Each domain $\mathbf{X}$ of values is evaluated with respect to a function $\mathbf{F}$: if True (white regions), the process retains $\mathbf{X}$, if False (dark grey regions) the process discounts $\mathbf{X}$, and if Unknown (light grey regions) $\mathbf{X}$ is recursively split and re-evaluated.

Despite their computational expense interval approaches have a number of valuable properties:

- an approximation of the solution set is guaranteed;
- each region $\mathbf{X}$ that is evaluated as True contains only solutions (i.e. each floating-point number in $\mathbf{X}$ is a solution);
- if all regions are False, the problem is guaranteed to have no solutions;
- linear, polynomial, non-polynomial and non-linear equations and inequalities can all be addressed.

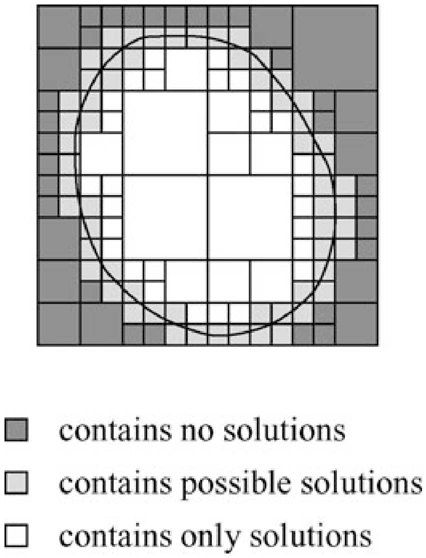


Figure 9: *Split and evaluation approach to solve $f(\mathbf{x}) \leq 0$ with interval arithmetic. Locations in white areas completely satisfy the relation, locations in dark areas do not satisfy the relation and grey areas contain locations that both satisfy and do not satisfy the relation.*

Jardillier *et al.* [JL98] proposed a constraint-based approach in which the path of the camera is created by declaring a set of properties on the desired shot including the vantage angle, framing and size of objects. The use of pure interval methods to compute camera paths in *The Virtual Cameraman* yields sequences of images fulfilling temporally indexed image properties. The path is modelled using a parameterized function (of degree 3) for each degree of freedom of the camera.

Christie *et al.* report a number of enhancements including more expressive camera path constraints and the integration of propagation techniques in the solving process [CLG02]. The path of the camera is modelled as a set of primitive camera movements sequentially linked together. These primitives are in essence cinematographic notions and include travellings, panoramics, arcings and any composition of these primitives. Unlike most approaches, which only guarantee the correctness of user-defined properties for a set of points on a camera trajectory (generally the start and end points plus some key points taken on the camera path), the interval-based approach guarantees the fulfilment of properties throughout the entire duration of the shot. Each primitive, referred to as a hypertube, is then treated as a separate, but related, constraint satisfaction problem. The solution technique processes the problem sequentially, constraining the end of hypertube i to join the beginning of hypertube $i + 1$.

Christie *et al.* replaced the evaluation process in Jardillier's evaluate-and-split approach with a pruning process based on local consistency techniques and propagation methods. Despite their guarantee of completeness these approaches fail to identify the sources of inconsistencies (e.g. incompatible constraints). Thus, the user must remove some constraints until a solution can be found. The constraints community offer some techniques to manage over-constrained problems, but only at a significant computational cost [JFM96].

Hierarchical constraint approaches are able to relax some of the constraints to provide an approximate solution to the problem. Bares *et al.* proposed CONSTRAINTCAM, a partial constraint satisfaction system for camera control [BGL98]. Inconsistencies are identified by constructing an *incompatible constraint pair graph.* If CONSTRAINTCAM fails to satisfy all the constraints of the original problem it relaxes weak constraints, and, if necessary, decomposes a single shot problem to create a set of camera placements that can be composed in multiple viewports [BL99]. CONSTRAINTCAM utilizes a restricted set of cinematographic properties (viewing angle, viewing distance and occlusion avoidance) and the constraint satisfaction procedure is applied to relatively small problems (i.e. involving two objects). Partial constraint satisfaction [Hos04] also requires the user to specify a hierarchy of constraints (ordering constraints on the visual properties of a shot).

5.4. Constrained Optimization Approaches

Pure optimization techniques enable the computation of a *possibly good* solution in that each property is satisfied to some degree. In the worst case, this partial satisfaction can lead to a contrived solution in which either one function dominates, or both are poorly satisfied. By contrast, pure complete constraint-based techniques compute the whole set of solutions, but often at considerable computational expense. In addition, pure constraint-based systems cannot yield approximate solutions for over-constrained problems. An acceptable compromise can be found in *constrained-optimization* approaches where the camera control problem is modelled as a set of properties to be enforced (the constraints) and a set of properties to be optimized (fitness functions to maximize).

In contrast to their early attempts at procedural camera control [DGZ92], Drucker and Zeltzer proposed the CAMDROID system which specifies behaviours for virtual cameras in terms of task-level goals (objective functions) and constraints on the camera parameters. They collect some primitive constraints into *camera modules,* which provide a higher level-means of interaction for the user. The *CFSQP* (Feasible Sequential Quadratic Programming) package is used to solve the resulting constrained optimization problem. Constraint functions and fitness functions in CAMDROID must be continuously differentiable. Thus their approach is limited to smooth functions subject to smooth constraints, conditions which are difficult to guarantee in many applications. Furthermore, the method is prone to local minima and is sensitive to the initial conditions, as with most local optimization techniques.

Combining constraint-based and optimization-based techniques yields more practical solutions to declarative camera control as both requirements (constraints) and preferences (fitness functions) can be addressed. In some approaches, the constraints can be solved by relying on geometric operators that efficiently reduce the search space before applying optimization techniques.

As a first step, geometric volumes can be computed solely based on camera positions, to build models of the feasible space [BMBT00, Pic02]. Classical stochastic search (as in [Pic02]), or heuristic search (as in [BMBT00]) can then be applied within the resulting volumes. Christie and Normand develop this notion to provide a semantic characterization of space in terms of cinematographic properties of each volume. The constraints are represented as *semantic volumes*, i.e. volumes that encompass regions of the space that fulfil a cinematographic property. This work can be considered as an extension of *visual aspects* [KvD79] and closely related to *viewpoint space partitioning* [PD90] for object recognition in computer vision.

In *visual aspects* all the viewpoints of a single polyhedron that share similar topological characteristics in the image are enumerated. A change of appearance of the polyhedron, with changing viewpoint, partitions the search space. Computing all the partitions enables the construction of regions of constant aspect (*viewpoint space partitions*). Christie *et al.* extend viewpoint space partitions to multiple objects and replace the topological characteristics of a polyhedron with properties such as occlusion, relative viewing angle, distance and relative object location. A *semantic volume* is then defined as a volume of possible camera locations that give rise to qualitatively equivalent shots with respect to cinematographic properties (see Figure 10).

Since a problem is described as a conjunction of properties, the volumes can be intersected to yield a solution. On identification of a solution, a numerical step is required to compute the orientation parameters of the camera (θ, ϕ and ψ) since the geometrical step only reduces search space for the position of the camera (x, y and z parameters). An optimization process based upon a continuous extension of the stochastic local search framework is applied in the semantic volume. Given an initial guess, each iteration of the algorithm generates a set of neighbours (i.e. small randomized modifications of the camera parameters) and selects the fittest parameter setting as the starting point for the next iteration. The complete characterization of the search space in terms of semantic volumes can be used as the foundation for interactive user navigation processes that explore and interact with the different classes of solutions. Due to the complexity in intersecting the volumes, this approach is inherently restricted to static environments.

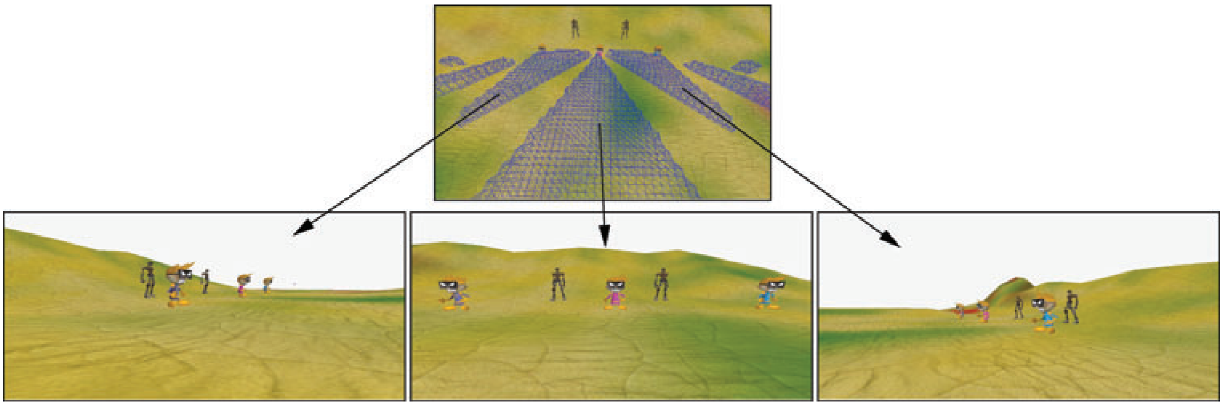


Figure 10: *Viewpoint space partitions corresponding to unoccluded views of multiple characters [CN05].*

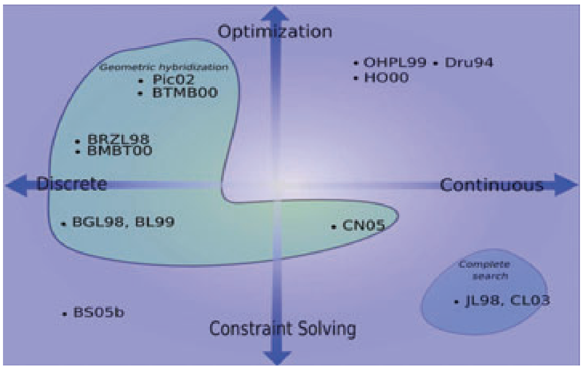


Figure 11: *Classification of declarative approaches: from discrete to continuous techniques (horizontal axis) and from constraint-based to optimization techniques (vertical axis).*

In summary, it is clear that a range of solution techniques are available, and solvers differ as to how they manage over-constrained and under-constrained problem formulations – both in their complete or incomplete search capabilities and their discretization of continuous camera domains. Consider the 2-dimensional classification of the approaches as presented in Figure 11. The horizontal axis corresponds to the nature of the domains and spans both fully discrete and fully continuous approaches. Discrete approaches rely on testing a subset of camera configurations through a regular or stochastic subdivision of the domains, with a view to reducing complexity of the $7 - dof$ search space. By contrast, continuous approaches provide techniques to explore all configurations. The nature of the solving technique is given by the vertical axis, and ranges from pure optimization techniques to pure constraint-based techniques. At one extreme, pure optimization techniques are considered as soft solving techniques for which the best (possibly local) solution is computed with respect to a function that is a measure of the violation of each property. At the other extreme, pure constraint satisfaction techniques can be considered as hard solving techniques. Such approaches perform an exhaustive exploration of the search space, thus providing the user with the set of solutions to the problem, or a guarantee that the problem has no solution.

6. Occlusion

Regardless of the application domain, a fundamental challenge for camera control approaches is the maintenance of unoccluded views. For example, computer games and animation require unoccluded views of principal characters most of the time. Likewise, scientific and information visualization must facilitate the maintenance of unoccluded views of visual features corresponding to important data points. Occlusion occurs when an object of interest is hidden by another object (or a set of objects) in the scene. Occlusion can be expressed as a continuous shot property from fully occluded to fully visible. Indeed, in particular domains, where the relative positions of objects are important, the controlled maintenance of partial occlusion is a desirable means of illustrating the relative depths of objects in a scene.

Camera control applications vary in the degree and character of their management of occlusion. Some approaches do address partial occlusion [CN05] and may offer a quantification of the occlusion as a percentage [BMBT00], or as pixel overlap counts [HO00], though most only consider total occlusion.

Dynamic environments require the ability to express changes in occlusion over time as is the case in many real-world spatial contexts. Indeed, when a character enters a car or building, it is often important to ensure that the end of the shot includes momentary occlusion of the actor by the doorway of an entrance, and likewise, the subsequent shot might typically commence with an occluded view of the character as he emerges from the doorway. The range of expressiveness required (in relation to occlusion) has a significant impact on

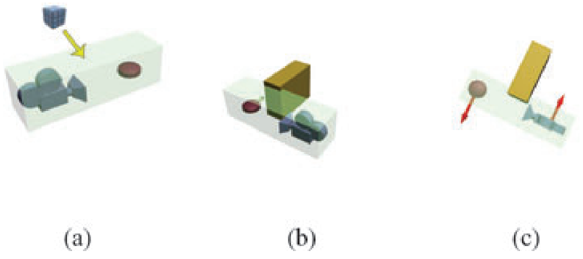


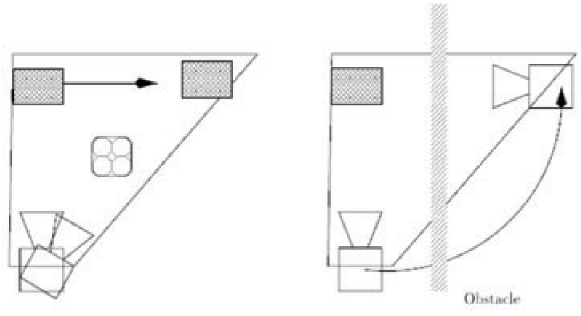
Figure 12: *Occlusion avoidance based on a bounding volume around both the camera and the target object: (a) due to a moving occluder; (b) due to the target motion; (c) due to the camera motion.*

its representation and computation. For dynamic scenes, approaches to occlusion management vary accordingly to the level of nondeterminism of the environment and *a priori* knowledge of the scene. Accordingly techniques can be divided into two classes: reactive or deliberative.

Reactive management of occlusion may be achieved by ray casts from the camera to the object of interest (moving the camera to avoid occlusions). The efficiency and simplicity of ray casting techniques make them the default choice for real-time applications, in particular, for computer games [Gio04] and game-based environments [BZRL98]. To improve performance, ray-object intersection can be replaced by a ray-bounding volume intersection. The choice of the bounding model (e.g. bounding sphere, bounding box, OBBtree) and the number of rays used, determine the precision and cost of the occlusion test [FvDFH90].

Due to its discrete and approximate nature, ray casting techniques are inherently incomplete. Occlusion can also be addressed in real-time contexts using *consistent regions* of space [BL99] using local spherical coordinates centered on the object of interest. The consistent region satisfying the occlusion for an object S is computed by projecting the bounding boxes of nearby potentially occluding geometry onto a discretized sphere surrounding S [BL99, PBG92, DZ95]. These projections are converted into global spherical coordinates and then negated to represent occlusion-free viewpoints on S.

Similarly, Courty [CM01] and Marchand [MH98, MC02] avoid occlusion in a target-tracking problem by computing an approximate bounding volume around both the camera and the target. Objects (i.e. not the camera or the target objects) are prevented from entering the volume corresponding to target motion (Figure 12(b)), camera motion (Figure 12(c)) or motion of other object (Figure 12(a)). This notion has been extended to address objects with unknown trajectories through the computation of approximate predictive volumes based on the current position and motion of an object (see Figure 13).



(a) Detection of a future occlusion.

(b) Detection of a future collision.

Figure 13: *Temporal extension of bounding volumes for collision/occlusion detection: the motion of both the camera and the object is enclosed in a volume and checked for intersection (see [MC02]).*

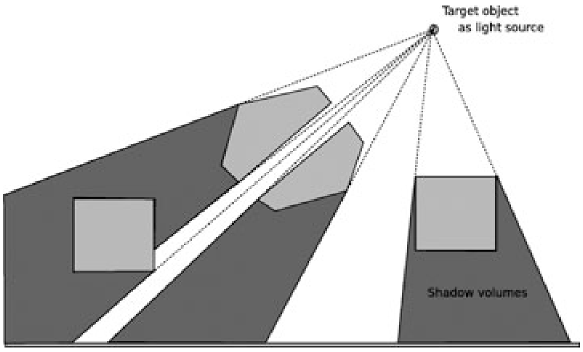


Figure 14: *The shadow volumes principle for computing occlusion-free locations. The target object is considered as a point-light source and the shadow partitions the search space into occluded and unoccluded regions.*

Other techniques [HO00, HHS01] take advantage of graphics hardware to address occlusion. By rendering the scene in hardware stencil buffers with a colour associated to each object, the number and extent of occluding objects can be very efficiently evaluated. Hardware rendering techniques have a number of attractive characteristics:

- the use of low-resolution buffers is often sufficient to deal with occlusions;
- hardware-based techniques are independent of the internal representation of the objects (e.g. no requirement for bounding volumes or approximations of the object);
- rendering the scene without the use of volume abstractions ensures occlusion is more accurately estimated.

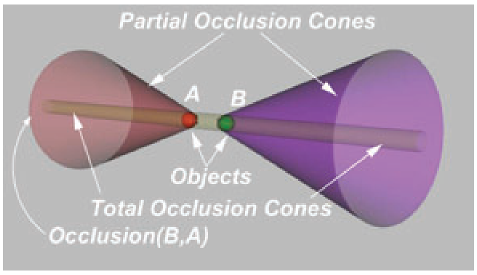


Figure 15: *Total and partial occlusion cones as presented in [CN05].*

Where performance is not a significant issue, occlusions can be addressed using object-space techniques. For example, a shadow-volume-based algorithm applied on the camera and the obstacles can be utilized to determine the parts of the scene that are unoccluded from the current viewpoint [Pic02] (cf. Figure 14). The object of interest is treated as the light source and the computed volumes represent the occluded areas. Geometric management of occlusion has also been proposed using occlusion cones computed from the bounding spheres of the scene elements [CN05]. This method distinguishes between partial and total occlusions (cf. Figure 15) but is only applicable to static scenes since each movement of an object requires the expensive recomputation of the cones.

7. Conclusion

Despite the variety of techniques for camera control, and attempts to deploy them across a broad range of applications, a number of key issues can be identified that cut across the landscape of existing work and will hopefully serve to define research into future camera control systems.

The management of complex 3D scenes almost inevitably requires us to use abstractions of the underlying geometry. To this end, existing practice typically utilizes simple primitives: points or bounding volumes such as spheres. Some proposals do consider precise geometry for occlusion purposes [HHS01, Pic02], but at a significant computational cost. Improving the quality of abstraction of the objects is a difficult but necessary enterprise. The principal constraints on the abstraction to be used is the solving process itself and the computational resources available. In existing work abstractions are either based on geometric simplifications to reduce the computational cost (from points to single bounding and hierarchical regions), or on semantic abstractions (object, character) to aid description. Ray-tracing techniques only provide a partial detection of occlusion that can lead to either over- or under-reactive cameras. Indeed, complex (but common) configurations, such as filming a character through the leaves of a tree, will require new approaches to the management of partial occlusion, incorporating criteria such as recognizability and temporal coherence.

Another pressing concern is the extension of the properties offered to the user, that is, the declarative power of a camera control framework. Although expressiveness is strongly related to both the underlying application and the chosen solving technique, it can be explored by reference to four criteria: the range of properties; the nature of the properties; the required level of abstraction of the 3D scene; and the extensibility. Properties can be specified on the camera parameters directly, on the path of the camera, or on the content of the projected image. Properties relative to the camera constrain (more or less directly) the intrinsic and extrinsic parameters of the camera. This encompasses properties such as preset focal lengths, high or low vantage angles that constrain the camera orientation, distances from camera to given objects and collision avoidance. Most contributions consider vantage angles and distances to the camera as important features [Dru94, HHS01, CL03, BTMB00], whereas collision is rarely managed, with the exception of some planning techniques (e.g. potential fields [Bec02] or probabilistic roadmaps [SGLM03]).

The ability to specify properties on the path of a camera is necessary to manage low-level issues such as collisions and path coherency [HHS01, HO00, CL03, NO03]. Although primarily related to reactive environments that require some predictive approaches to enforce smoothness [HHS01, MC02], path coherency aims to smooth out the camera path where sudden changes of translational or rotational velocities occur. Higher-level problems include the cinematographic nature of the path [CL03, CLDM03] and the narrative goals [HCS96, BL97] and their solution is likely to form the basis of more engaging and cinematic experiences, both in interactive and automated approaches.

Indeed, properties need to be specified at a (higher) semantic level, i.e. based on the nature of the functionality and the relationship between the objects. Research to date has enumerated a large set of screen properties, from low-level *object-is-on-the-screen* enforcement to more sophisticated image composition rules (see e.g. [GRMS01]). Most consider purely geometric properties, e.g. where to project 3D points or objects on the screen and how to organise and lay them out. This encompasses absolute location of objects [Bli88, Dru94, HCS96, HHS01, GW92], framing of objects by constraining them in a given area (in or out of the screen) [JL98, BMBT00, OHPL99] and relative location (e.g. *A* is on the left of *B*) [OHPL99]. However, cinematography provides a detailed vocabulary that has been only partially transposed to virtual camera control. Shot lexicons, incorporating elements such as the *over-the-shoulder* and *American Shot* (shoot a character above the knees), have been used both in a number of automated approaches based on idioms ([CAH*96, HCS96]) and in declarative approaches which in turn propose modelling languages [HO00, OHPL99, JL98]. The successful exploitation of such a lexicon requires the augmentation of geometric data with semantic information relative to the nature, role, relations and intents of characters and objects in the narrative.

Finally, we can think of properties in terms of both aesthetics and cognition. Gooch *et al.* [GRMS01] compute slight modifications of point of views in order to satisfy some high-level composition rules (such as the rules of thirds and fifths), but similarly only consider the geometry of the objects. Colour, lighting and semantics are essential in considering aesthetic goals such as balance, unity and composition. A number of approaches characterise object recognizability and provide techniques to compute views that maximise the visual saliency for static scenes [HHO05]. Cognitive levels have been addressed both by the robotics and the computer graphics communities whereby a minimal set of viewpoints (canonical views) for recognizing an object and/or detecting its features is computed. The principal metrics are view likelihood, view stability [WW97] and view goodness [BS05a] which have been computed using entropy measures [VFSH01]. Extensions to automated navigation have also been explored for historical data visualization [SS02] and scene understanding [SP05]. Finally, notions such as the emotional response that a view can evoke has been largely ignored although Tomlinson *et al.* propose camera rules to cover a set of communicative goals based on the emotion to be conveyed [TBN00].

While some approaches utilize a fixed set of properties [HCS96, HHS01] a number of contributions have concentrated on providing frameworks to allow the addition of new properties. Most automated approaches (see Section 5.1) rely on solving techniques that propose extensible frameworks. By simply declaring new algebraic relations between the unknowns, the constraint-based techniques offer a natural way to extend expressiveness, though usually at the cost of efficiency and without techniques to properly manage over-constrained systems. On the other hand, optimization-based techniques offer a similar extensibility, but require hand tuning to adapt the weight related to the new properties. Future systems need to build on expressive frameworks that incorporate temporal aspects in solving as well as hybrid discrete/continuous system management and effective means to deal with over-constrained systems.

This review of the state-of-the-art has presented an overview of camera control in computer graphics from interactive techniques to completely automated camera systems. An analysis of cinematic and photographic practice has helped us to develop a classification of the expressiveness of camera shots, ranging from geometric to perceptual and aesthetic. This classification was undertaken with respect to both the geometric expressiveness of the approaches and the solution mechanisms, from interactive mappings of user inputs, to automatic reactive camera control. What has become clear is that the next generation of camera control systems requires not only efficient implementations but also empirical work on the characterization of higher-level properties that will facilitate the maintenance of aesthetic and emotionally evocative views. Whilst the aesthetic properties are likely to be founded on an adequate cognitive model, work on exploiting editing rules to effectively engage the user are still very much in their infancy [FF04, TBN00]. We see this as a key area for interdisciplinary research by both computer scientists and cognitive psychologists, not only to the benefit of computer graphics, but as a window on the nature of cognition itself.

References

[Ari76] Arijon D.: *Grammar of the Film Language*. Hastings House Publishers, 1976.

[AVF04] Andújar C. G., Vázquez P. P. A., Fairén M. G.: Way-finder: Guided tours through complex walkthrough models. *Proceedings of the Eurographics 2004 Conference (EG 04) 23*, 3 (2004), 499–508.

[Bar07] Bares W.: A photographic composition assistant for intelligent virtual 3d camera systems. In *Proceedings of Smartgraphics 2007 (SG 07)* (2007), Springer, pp. 172–183.

[BDSG04] Byers Z., Dixion M., Smart W. D., Grimm C. M.: Say cheese! experiences with a robot photographer. *AI Mag. 25*, 3 (2004), 37–46.

[Bec02] Beckhaus S.: *Dynamic Potential Fields for Guided Exlporation in Virtual Environments*. PhD thesis, Fakultät für Informatik, University of Magdeburg, 2002.

[BGL98] Bares W. H., Gregoire J. P., Lester J. C.: Realtime Constraint-Based cinematography for complex interactive 3D worlds. In *Proceedings of AAAI-98/IAAI-98* (1998), pp. 1101–1106.

[BJH01] Bowman D. A., Johnson D. B., Hodges L. F.: Testbed evaluation of virtual environment interaction techniques. *Presence: Teleoper. Virtual Environ. 10*, 1 (2001), 75–95.

[BKF*02] Burtnyk N., Khan A., Fitzmaurice G., Balakrishnan R., Kurtenbach G.: Stylecam: Interactive stylized 3d navigation using integrated spatial & temporal controls. In *Proceedings of the ACM symposium on User interface software and technology (UIST 02)* (New York, NY, USA, 2002), ACM, pp. 101–110.

[BL97] Bares W. H., Lester J. C.: Cinematographic user models for automated realtime camera control in dynamic 3D environments. In *Proceedings of the sixth International Conference on User Modeling (UM 97)* (Vien New York, 1997), Springer-Verlag, pp. 215–226.

[BL99] Bares W. H., Lester J. C.: Intelligent multi-shot visualization interfaces for dynamic 3D worlds. In

Proceedings of the 4th international conference on Intelligent user interfaces (IUI 99) (New York, NY, USA, 1999), ACM Press, pp. 119–126.

[Bli88] BLINN J.: Where am I? what am I looking at? *IEEE Computer Graphics and Applications* (July 1988), 76–81.

[BMBT00] BARES W., MCDERMOTT S., BOUDREAUX C., THAINIMIT S.: Virtual 3D camera composition from frame constraints. In *Proceedings of the ACM international conference on Multimedia (ACMMM 00)* (2000), ACM Press, pp. 177–186.

[BRZL98] BARES W. H., RODRIGUEZ D. W., ZETTLEMOYER L. S., LESTER J. C.: Task-sensitive cinematography interfaces for interactive 3d learning environments. In *Proceedings Fourth International Conference on Intelligent User Interfaces (IUI 98)* (1998), pp. 81–88.

[BS05a] BORDOLOI U., SHEN H.-W.: View selection for volume rendering. In *Proceedings of the IEEE Visualization Conference (VIS 2005)* (2005), IEEE Computer Society, p. 62.

[BS05b] BOURNE O., SATTAR A.: Applying constraint weighting to autonomous camera control. In *Proceedings of Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2005)* (2005), pp. 3–8.

[BST05] BALET O., SUBSOL G., TORGUET P.: *Virtual Storytelling, Using Virtual Reality Technologies for Storytelling, Third International Conference (ICVS 05)*. Springer-Verlag, Strasbourg, France, December 2005.

[BTMB00] BARES W., THAINIMIT S., MCDERMOTT S., BOUDREAUX C.: A model for constraint-based camera planning. In *Smart Graphics AAAI Spring Symposium* (Stanford, California, March 2000), pp. 84–91.

[But97] BUTZ A.: Animation with CATHI. In *Proceedings of American Association for Artificial Intelligence (IAAI 97)* (1997), AAAI Press, pp. 957–962.

[BZRL98] BARES W. H., ZETTLEMOYER L. S., RODRIGUEZ D. W., LESTER J. C.: Task-sensitive cinematography interfaces for interactive 3D learning environments. In *Intelligent User Interfaces (IUI 98)* (1998), pp. 81–88.

[CAH*96] CHRISTIANSON D. B., ANDERSON S. E., HE L., SALESIN D. H., WELD D. S., COHEN M. F.: Declarative Camera Control for Automatic Cinematography. In *Proceedings of the American Association for Artificial Intelligence (AAAI 96)* (1996), pp. 148–155.

[CHL*98] CHIOU R., HAUFMAN A., LIANG Z., HONG L., ACHNIOTOU M.: Interactive path planning for virtual endoscopy. In *Proceedings of the Nuclear Science and Medical Imaging Conference* (1998), pp. 2069–2072.

[CL03] CHRISTIE M., LANGUÉNOU E.: A constraint-based approach to camera path planning. In *Proceedings of the Third International Symposium on Smart Graphics (SG 03)* (2003), vol. 2733, Springer, pp. 172–181.

[CLCM02] CHARLES F., LUGON J., CAVAZZA M., MEAD S.: Real-time camera control for interactive storytelling. In *Game-On: International Conference for Intelligent Games and Simulations* (London, UK, 2002).

[CLDM03] COURTY N., LAMARCHE F., DONIKIAN S., MARCHAND E.: A cinematography system for virtual storytelling. In *International Conference on Virtual Storytelling (ICVS 03)* (Toulouse, France, November 2003), pp. 30–34.

[CLG02] CHRISTIE M., LANGUÉNOU E., GRANVILLIERS L.: Modeling camera control with constrained hypertubes. In *Proceedings of the Constraint Programming Conference (CP 02)* (2002), vol. 2470, Springer-Verlag, pp. 618–632.

[CM01] COURTY N., MARCHAND E.: Computer animation: A new application for image-based visual servoing. In *Proceedings of International Conference on Robotics and Automation, (ICRA 01)* (2001), pp. 223–228.

[CMS88] CHEN M., MOUNTFORD S., SELLEN A.: A study in interactive 3d rotation using 2d input devices. In *Proceedings of the Siggraph Conference* (Aug. 1988), vol. 22-4, ACM Computer Graphics, pp. 121–130, 1998.

[CN05] CHRISTIE M., NORMAND J.-M.: A semantic space partitioning approach to virtual camera control. In *Proceedings of the Eurographics Conference (EG 2005)* (2005), vol. 24, Computer Graphics Forum, pp. 247–256.

[DGZ92] DRUCKER S. M., GALYEAN T. A., ZELTZER D.: Cinema: A system for procedural camera movements. In *Proceedings of the 1992 symposium on Interactive 3D graphics (SI3D 92)* (New York, NY, USA, 1992), ACM Press, pp. 67–70.

[DH02] DENNIS B. M., HEALEY C. G.: Assisted navigation for large information spaces. In *Proceedings of the conference on Visualization (Vis 02)* (Washington, DC, USA, 2002), IEEE Computer Society, pp. 419–426.

[Dru94] DRUCKER S. M.: *Intelligent Camera Control for Graphical Environments*. PhD thesis, School of Architecture and Planning, Massachusetts Institute of Technology MIT Media Lab, 1994.

[DZ94] DRUCKER S. M., ZELTZER D.: Intelligent camera control in a virtual environment. In *Proceedings of Graphics Interface (GI 94)* (Banff, Alberta, Canada, 1994), pp. 190–199.

[DZ95] DRUCKER S. M., ZELTZER D.: Camdroid: A System for Implementing Intelligent Camera Control. In

Proceedings of the 1995 symposium on Interactive 3D graphics (SI3D 95) (1995), pp. 139–144.

[ECR93] ESPIAU B., CHAUMETTE F., RIVES P.: A new approach to visual servoing in robotics. In *Selected Papers from the Workshop on Geometric Reasoning for Perception and Action* (London, UK, 1993), Springer-Verlag, pp. 106–136.

[FCOL00] FLEISHMAN S., COHEN-OR D., LISCHINSKI D.: Automatic camera placement for image-based modeling. *Computer Graphics Forum 19*, 2 (2000), 101–110.

[FF04] FRIEDMAN D. A., FELDMAN Y. A.: Knowledge-Based Cinematography and Its Applications. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 04)* (2004), IOS Press, pp. 256–262.

[FPB87] FREDERICK P. BROOKS J.: Walkthrough—a dynamic graphics system for simulating virtual buildings. In *SI3D '86: Proceedings of the 1986 workshop on Interactive 3D graphics* (New York, NY, USA, 1987), ACM, pp. 9–21.

[FvDFH90] FOLEY J. D., VAN DAM A., FEINER S. K., HUGHES J. F.: *Computer Graphics: Principles and Practice*, 2nd ed. Addison-Wesley Publishing Co., Reading, MA, 1990.

[Gio04] GIORS J.: The full spectrum warrior camera system. In *Game Developers Conference (GDC 04)* (2004).

[GRMS01] GOOCH B., REINHARD E., MOULDING C., SHIRLEY P.: Artistic composition for image creation. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques* (London, UK, 2001), Springer-Verlag, pp. 83–88.

[GW92] GLEICHER M., WITKIN A. P.: Through-the-lens camera control. In *Proceedings of the Siggraph Conference* (1992), ACM Computer Graphics, pp. 331–340.

[HCS96] HE L., COHEN M. F., SALESIN D. H.: The virtual cinematographer: A paradigm for automatic real-time camera control and directing. In *Proceedings of the Siggraph Conference* (Aug. 1996), ACM Computer Graphics, pp. 217–224.

[HHO05] HOWLETT S., HAMILL J., O'SULLIVAN C.: Predicting and evaluating saliency for simplified polygonal models. *ACM Transactions on Applied Perception 2*, 3 (2005), 286–308.

[HHS01] HALPER N., HELBING R.,STROTHOTTE T.: A camera engine for computer games: Managing the trade-off between constraint satisfaction and frame coherence. In *Proceedings of the Eurographics Conference (EG 2001)* (2001), vol. 20, Computer Graphics Forum, pp. 174–183.

[HMK*97] HONG L., MURAKI S., KAUFMAN A., BARTZ D., HE T.: Virtual voyage: interactive navigation in the human colon. In *Proceedings of the Siggraph Conference* (New York, NY, USA, 1997), ACM Computer Graphics, pp. 27–34.

[HO00] HALPER N., OLIVIER P.: CAMPLAN: A Camera Planning Agent. In *Smart Graphics AAAI Spring Symposium* (March 2000), pp. 92–100.

[Hos04] HOSOBE H.: Hierarchical nonlinear constraint satisfaction. In *Proceedings of the ACM symposium on Applied computing (SAC 04)* (New York, NY, USA, 2004), ACM Press, pp. 16–20.

[HW97] HANSON A., WERNERT E.: Constrained 3d navigation with 2d controllers. In *Proceedings of the IEEE Visualization Conference (VIS 97)* (1997), pp. 175–182.

[JFM96] JAMPEL M., FREUDER E. C., MAHER M. J. (Eds.):. *Over-Constrained Systems* (1996), vol. 1106, Springer.

[JL98] JARDILLIER F., LANGUÉNOU E.: Screen-space constraints for camera movements: the virtual cameraman. In *Proceedings of the Eurographics Conference (EG 98)* (1998), vol. 17, Blackwell Publishers, pp. 175–186.

[JPK98] JUNG M.-R., PAIK D., KIM D.: A camera control interface based on the visualization of subspaces of the 6d motion space of the camera. *Proceedings of the Pacific Graphics Conference (PG 98)* (1998), 198.

[Kat91] KATZ S.: *Film Directing Shot by Shot: Visualizing from Concept to Screen*. Michael Wiese Productions, 1991.

[Kha86] KHATIB O.: Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research 5*, 1 (1986), 90–98.

[KHRO01] KOWALSKI M. A., HUGHES J. F., RUBIN C. B., OHYA J.: User-guided composition effects for art-based rendering. In *Proceedings of the symposium on Interactive 3D graphics (I3D 01)* (New York, NY, USA, 2001), ACM, pp. 99–102.

[KK88] KAMADA T., KAWAI S.: A simple method for computing general position in displaying three-dimensional objects. *Computer Vision Graphics and Image Processing 41*, 1 (1988), 43–56.

[KKH95] KUNG M. H., KIM M. S., HONG S.: Through-the-lens camera control with a simple jacobian matrix. In *Proceedings of Graphics Interface (GI 95)* (1995), pp. 117–178.

[KKS*05] KHAN A., KOMALO B., STAM J., FITZMAURICE G., KURTENBACH G.: Hovercam: interactive 3d navigation for proximal object inspection. In *Proceedings of the 2005 symposium on Interactive 3D graphics and games (SI3D 05)* (New York, NY, USA, 2005), ACM Press, pp. 73–80.

[KvD79] Koenderink J., Van Doorn J.: The internal representation of solid shape with respect to vision. *Biological Cybernetics 32* (1979), 211–216.

[LFN04] Lok S., Feiner S., Ngai G.: Evaluation of visual balance for automated layout. In *Proceedings of the international conference on Intelligent user interfaces (IUI 04)* (New York, NY, USA, 2004), ACM, pp. 101–108.

[LH04] Li T.-Y., Hsu S.-W.: An intelligent 3d user interface adapting to user control behaviors. In *Proceedings of the 9th International Conference on Intelligent User Interface (IUI 04)* (New York, NY, USA, 2004), ACM Press, pp. 184–190.

[LT00] Li T.-Y., Ting H.-K.: An intelligent user interface with motion planning for 3d navigation. In *Proceedings of the Virtual Reality International Conference (VR 00)* (2000), p. 177.

[LX05] Li T.-Y., Xiao X.-Y.: An interactive camera planning system for automatic cinematographer. In *Proceedings of the 11th International Multimedia Modelling Conference (MMM'05)* (Washington, DC, USA, 2005), IEEE Computer Society, pp. 310–315.

[Mas65] Mascelli J.: *The Five C's of Cinematography: Motion Picture Filming Techniques*. Cine/Grafic Publications, Hollywood, 1965.

[MC02] Marchand E., Courty N.: Controlling a camera in a virtual environment. *The Visual Computer Journal 18*, 1 (2002), 1–19.

[MCR90] Mackinlay J. D., Card S. K., Robertson G. G.: Rapid Controlled Movement Through a Virtual 3D Workspace. *Computer Graphics 24*, 4 (Aug. 1990), 171–176.

[MH98] Marchand E., Hager G.: Dynamic sensor planning in visual servoing. In *Proceedings of the International Conference on Robotics and Automation (ICRA 98)* (Leuven, Belgium, May 1998), vol. 3, pp. 1988–1993.

[Moo66] Moore R.: *Interval Analysis*. Prentice Hall, 1966.

[NLK93] Nodinem C., Locher J., Krunpinski E.: *The role of formal art training on perception and aesthetic judgement of art compositions*. Leonardo, 1993.

[NO03] Nieuwenhuisen D., Overmars M. H.: *Motion Planning for Camera Movements in Virtual Environments*. Tech. Rep. UU-CS-2003-004, Institute of Information and Computing Sciences, Utrecht University, 2003.

[OHPL99] Olivier P., Halper N., Pickering J., Luna P.: Visual Composition as Optimisation. In *AISB Symposium on AI and Creativity in Entertainment and Visual Art* (1999), pp. 22–30.

[Pal96] Palamidese P.: A camera motion metaphor based on film grammar. *Journal of Visualization and Computer Animation 7*, 2 (1996), 61–78.

[PBG92] Phillips C. B., Badler N. I., Granieri J.: Automatic viewing control for 3d direct manipulation. In *Proceedings of the symposium on Interactive 3D graphics (SI3D 92)* (1992), ACM Press New York, NY, USA, pp. 71–74.

[PD90] Plantinga H., Dyer C. R.: Visibility, Occlusion, and the aspect graph. *International Journal of Computer Vision 5*, 2 (Nov 1990), 137–160.

[Pic02] Pickering J. H.: *Intelligent Camera Planning for Computer Graphics*. PhD thesis, Department of Computer Science, University of York, 2002.

[SF91] Seligmann D. D., Feiner S.: Automated generation of intent-based 3d illustrations. In *Proceedings of the Siggraph Conference* (New York, NY, USA, 1991), ACM Computer Graphics, pp. 123–132.

[SF93] Seligmann D. D., Feiner S.: Supporting interactivity in automated 3d illustrations. In *Proceedings of the 1st international conference on Intelligent user interfaces (IUI 93)* (New York, NY, USA, 1993), ACM Press, pp. 37–44.

[SGLM03] Salomon B., Garber M., Lin M. C., Manocha D.: Interactive navigation in complex environments using path planning. In *Proceedings of the 2003 symposium on Interactive 3D graphics (SI3D 03)* (New York, NY, USA, 2003), ACM Press, pp. 41–50.

[Sho92] Shoemake K.: Arcball: a user interface for specifying three-dimensional orientation using a mouse. In *Proceedings of Graphics Interface (GI 92)* (May 1992), pp. 151–156.

[Sny92] Snyder J.: Interval analysis for computer graphics. In *Proceedings of the Siggraph Conference* (1992), vol. 22-4, ACM Computer Graphics, pp. 121–130.

[SP05] Sokolov D., Plemenos D.: Viewpoint quality and scene understanding. In *Proceedings of the Eurographics Symposium on Virtual Reality, Archaeulogy and Cultural Heritage (VAST 05)* (Pisa, Italy, 2005), Eurographics Association, pp. 67–73.

[SP06] Sokolov D., Plemenos D.: Methods and data structures for virtual world exploration. *The Visual Computer 22*, 7 (July 2006), 506–516.

[SS02] STOEV S. L., STRASSER W.: A case study on automatic camera placement and motion for visualizing historical data. In *Proceedings of the IEEE Visualization Conference (VIS 02)* (Washington, DC, USA, 2002), IEEE Computer Society, pp. 545–548.

[Stu99] STUERZLINGER W.: Imaging all visible surfaces. In *Proceedings of Graphics interface (GI 99)* (San Francisco, CA, USA, 1999), Morgan Kaufmann Publishers Inc., pp. 115–122.

[TBGT91] TURNER R., BALAGUER F., GOBBETTI E., THALMANN D.: Physically-based interactive camera motion control using 3D input devices. In *Scientific Visualization of Physical Phenomena* (1991), Springer Verlag, pp. 135–145.

[TBN00] TOMLINSON B., BLUMBERG B., NAIN D.: Expressive autonomous cinematography for interactive virtual environments. In *Proceedings of the Fourth International Conference on Autonomous Agents (AAMAS 00)* (Barcelona, Catalonia, Spain, 2000), ACM Press, pp. 317–324.

[TRC01] TAN D. S., ROBERTSON G. G., CZERWINSKI M.: Exploring 3d navigation: combining speed-coupled flying with orbiting. In *Proceedings of the ACM CHI conference on Human factors in computing systems (CHI 01)* (New York, NY, USA, 2001), ACM Press, pp. 418–425.

[VFSG06] VIOLA I., FEIXAS M., SBERT M., GRÖLLER E.: Importance-driven focus of attention. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 933–940.

[VFSH01] VÁZQUEZ P.-P., FEIXAS M., SBERT M., HEIDRICH W.: Viewpoint selection using viewpoint entropy. In *Proceedings of the Vision Modeling and Visualization Conference (VMV 01)* (2001), Aka GmbH, pp. 273–280.

[VFSH03] VÀZQUEZ P.-P., FEIXAS M., SBERT M., HEIDRICH W.: Automatic view selection using viewpoint entropy and its application to image-based modelling. *Computer Graphics Forum 22*, 4 (2003), 689–700.

[WFH*97] WOOD D. N., FINKELSTEIN A., HUGHES J. F., THAYER C. E., SALESIN D. H.: Multiperspective panoramas for cel animation. In *Proceedings of the Siggraph Conference* (New York, NY, USA, 1997), ACM Computer Graphics, pp. 243–250.

[WO90] WARE C., OSBORNE S.: Exploration and virtual camera control in virtual three dimensional environments. In *Proceedings of the 1990 symposium on Interactive 3D graphics (SI3D 90)* (New York, NY, USA, 1990), ACM Press, pp. 175–183.

[WW97] WEINSHALL D., WERMAN M.: On view likelihood and stability. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*, 2 (1997), 97–108.

[XH98] XIAO D., HUBBOLD R. J.: Navigation guided by artificial force fields. In *Proceedings of ACM CHI Conference on Human Factors in Computing Systems (CHI 98)* (1998), Wesley A., (Ed.), pp. 179–186.

[ZF99] ZELEZNIK R. C., FORSBERG A. S.: Unicam - 2d gestural camera controls for 3d environments. In *Proceedings of the symposium on Interactive 3D graphics (SI3D 99)* (1999), pp. 169–173.

[ZRS03] ZANCANARO M., ROCCHI C., STOCK O.: Automatic video composition. In *Proceedings of the Third International Symposium on Smart Graphics (SG 03)* (2003), pp. 192–201.

[ZSA03] ZANCANARO M., STOCK O., ALFARO I.: Using cinematic techniques in a multimedia museum guide. In *Proceedings of Museums and the Web* (March 2003).