



HAL
open science

Tabletop Interactive Camera Control

Mukesh Barange

► **To cite this version:**

| Mukesh Barange. Tabletop Interactive Camera Control. Graphics [cs.GR]. 2010. dumas-00530632

HAL Id: dumas-00530632

<https://dumas.ccsd.cnrs.fr/dumas-00530632>

Submitted on 29 Oct 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE



centre de recherche **RENNES - BRETAGNE ATLANTIQUE**



MASTER THESIS

Tabletop Interactive Camera Control

Author: Mukesh Barange

Supervisors: Marc Christie (INRIA Rennes)

Frederic Vernier (CNRS LIMSI), Rami Ajaj (CNRS LIMSI)

Laboratory: INRIA Rennes

Team: Project Bunraku

Master Research - Software and Formal Methods (SICH)

TELECOM Bretagne, Brest

21/06/2010

Title

Tabletop Interactive Camera Control

21/06/2010

Abstract

Multi-touch surface represents an emerging technology for display and interaction in 3D virtual environment. It can be extended for interactively controlling a virtual camera inspired with cinematographic primitives. We propose a frame-based interaction metaphor called frame-metaphor which allows to control and compose cinematographic primitives. Through this metaphor user can perform cinematographic operations by interacting with a rectangular frame on multi-touch surface. With reference to the parameters of the frame new camera parameters are computed. This technique allows the user to navigate freely in virtual environment. In this work we design different mappings between user inputs and camera parameters and describe how solution techniques for interactive camera control can be build upon through the lens camera techniques[8, 14].

Chapter 1

Introduction

1.1 General context

Computer graphics is concerned with modeling, rendering, animation and manipulation of 2D and 3D objects by users. In computer graphics, users perceive and interact with 3D virtual environments by viewing the scene through the “eyes” of a virtual camera. Therefore camera control is an essential component of many interactive 3D applications, such as navigation in a 3D environment, data visualization, virtual tours, virtual storytelling and 3D video games. Controlling virtual camera covers positioning, motion planning and computing the user’s viewpoint in a scene. The user normally has a clear vision of the result he/she seeks in terms of visual outputs. And the virtual camera acts as an intermediary between the production of 2D/3D scene and their mental interpretation. Therefore camera control in virtual environments plays an essential role in transmission of information.

To position and orient a camera in a 3D world is a difficult task. A number of scientific studies have been conducted to facilitate the camera control in virtual environments. *Automatic* approaches generally consider the specification of declarative characterization of properties on camera paths or camera shots by reference to which the values of the camera parameters are derived. *Interactive* approaches provide a correlation between user inputs and degrees of freedom (DOFs). These approaches propose a set of mappings between dimensions of the input devices (mouse, keyboard, joysticks, or touch-surfaces) and the camera parameters. But interactive camera control has received relatively little attention in 3D computer graphics. Interactively controlling a camera is a complex process. Users sometimes find it problematic to deal simultaneously with all seven DOFs (3DOFs for translation, 3DOFs for orientation, and 1DOF for scaling) in highly dynamic environments also the nature and the complexity of mappings depends upon target applications. Also, no perfect 7DOFs device is available to manipulate a camera. In fact, space-mouse has only 6DOFs, and it is manipulated with hands like an external object.

The interaction and manipulation of 3D objects are also two other areas

of re-looking for computer graphics. The industrial and commercial applications require specific manipulators dedicated to the modeling of 3D objects, in particular the handling of the mouse, joysticks and many other interaction devices such as multi-touch surfaces in three-dimensional environment. However, although a set of methods has been defined in this goal, the progress in development of new methods of interaction is slow.

In parallel, multi-touch surfaces are one of the central emerging technologies these days. They create workspaces suitable for a single user as well as for the collaboration between multiple users. Various kinds of tracking technologies, projection systems, and sophisticated interaction devices have been realized and combined into new groupware for single and multi-user environments. Multi-touch surfaces can accept natural hand gestures and other intuitive devices (e.g. mouse and space-mouse etc.) as input. The strengths of the gesture based interactions are the intuitive, consistent look and feel, and the direct engagement of the objects. Now the multi-touch surfaces like tabletop, PDA and many other mobile devices etc. offer an interesting alternative to control objects with many DOFs. Many 2D and 3D interaction techniques [7, 13, 19] that allow direct manipulation of objects, have shown the benefits of multi-touch and have shown to be very appealing to users.

However, there are many issues related to the interactive control of a virtual camera using multi-touch surfaces. This leads to the requirement of detecting the constraints on camera (height, distance or orientation relative to the objects in 3D world), mapping between user inputs to the appropriate camera parameters, identifying the appropriate gestures for camera control, handling collision and visibility of objects of interest etc. In this context, the relevance of multi-touch interaction as a controller of an interactive virtual camera needs to be studied and explored.

1.2 Objectives

The Objectives of the internship are to propose a model for computing the camera path interactively and identifying the mapping of natural hand gestures in ad-equation with the task, to the camera parameters. We propose a high-level interaction metaphor so called *frame metaphor*, in which the position and orientation of a camera is indirectly controlled by high level properties of the rectangular frame. This metaphor is different from other metaphors proposed by [23] (e.g. eyeball in hand, scene in hand, flying vehicle) that directly control the virtual camera. In contrast to these metaphors, our frame metaphor indirectly controls the position and orientation of the camera by allowing the user to manipulate the position and orientation of the frame, with reference to which new camera position is calculated. In this metaphor users can perform various cinematographic operations such as rotation, translation, zooming, tilting (and many other possible compositions of these) by interacting with a frame. We propose to directly control what we see in the environment by using this metaphor. We define a set of natural gestures for multi-touch

surfaces, therefore users can have up to 7DOFs to manipulate frame in 3D environment. Unlike other metaphors, our frame metaphor can be used for selection and manipulation of 3D objects along with indirect camera control.

This report is organized as follows: before presenting our contributions to the problem tabletop interactive camera control in virtual environments, we build a state of the art of existing methods for camera control and tabletop interactions in chapters 2 and 3 respectively. Our contribution concerning a camera control in virtual environment and mapping of hand gestures for tabletop interaction to camera parameters is presented in chapter 4 and then results and comparisons with other methods are presented. Aspects and conclusion is presented in chapter 5.

Chapter 2

State-of-the-art techniques for camera control in a virtual environment

2.1 General Presentation

Virtual Camera control is an important problem in a number of different 3D computer graphics areas. Many applications such as scientific visualization, animation and 3D games all make considerable use of virtual camera control in a three-dimensional environment. Camera control systems in many 3D computer applications such as virtual walk-through, virtual storytelling, 3D games etc. is inspired by cinematographic techniques. Many 3D applications require users to interact with the virtual world. Users interact with three-dimensional virtual environment by viewing the scene generated by the virtual camera. Good camera control and planning techniques can give users deeper feelings about 3D environment. A virtual 3D camera view is defined by parameters of camera position, direction vector, and field of view (lens angle). Camera control which encompasses viewpoint computing, and motion planning, is an integral part of any 3D interface. Numbers of techniques have been proposed for both interactive and automatic control of virtual camera. Computing the camera path, evaluation and avoidance of occlusions are some key challenges in interactive camera control. Many applications such as 3D games, visualization of complex multidimensional data in 3D environment, emotional expression of virtual characters in edutainment applications and character-based interactive storytelling approaches require efficient camera control in virtual environment. The basic objectives of the camera control are choosing the best position and angle of the camera and providing an occlusion free view of virtual environment.

Our presentation of the state of the art in interactive camera control reflects the progression from interactive direct approaches to interactive indirect computation and emphasizes the principal challenges for camera control, such as management of collision, occlusion and expressiveness. We begin with the

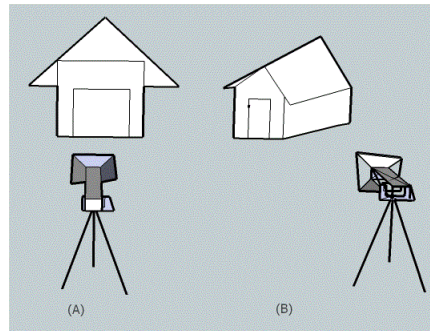


Figure 2.1: Shots illustrating how change in camera position can convey compositional parameters like depth/perspective

discussion of various camera configurations, followed by briefly describing cinematographic properties for camera control. Following sections are related to interactive camera control approaches including: a taxonomy of interactive camera control metaphors such as eyeball in hand, scene in hand etc. and then we will discuss in details, the interactive camera control techniques for exploring 3D objects, exploring complex environments and through the lens camera control techniques.

2.2 Camera planning in Cinematography

Cinematography is defined as the art of film-making. It provides guidelines for motion picture and how the camera in particular should be used in order to accomplish tasks such as engaging the interest of the viewer, and presenting the content in an interesting and coherent manner. Placement of camera relative to the subject, shot composition, and lighting arrangements are the principal issues that cinematography addresses.

2.2.1 Camera positioning:

Cinematography provides guidelines for positioning and movement of the camera. There are a number of different types of camera movements. Two of the main ones are tracking and panning. Tracking is when the camera moves alongside a character while filming them and panning is when it rotates on its vertical axis. Tracking can give the viewer the impression that they are walking alongside a character and panning is often used to take in the expanse of a setting. Also positioning the camera must address the issue of choosing the best position and angle for the camera out of multiple possible positions and angles, for instance, as shown in Fig. 2.1, the front view of a house does not convey the sense of perspective. Choosing a different angle conveys these properties more effectively.

Camera positioning ensures the general spatial arrangement of elements of the scene with respect to the camera, thereby imposing constraints on the composition of a shot. That is, the position of the camera determines the class of shot that is achievable. In cinematography, a *scene* is a single setting and its depiction that typically includes one or more characters whose actions are of interest. A *shot* is a continuous view filmed by one camera without interruption, which can be broadly classified accordingly to the amount of the subject included in the shot as: *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). The transition from one shot to the next is known as a *cut*.

Camera positioning and shot composition in complex environment are difficult tasks, therefore, the challenge for the camera planning is to formulate these cinematographic properties in a manner appropriate for the applications.

2.3 Camera control in computer graphics

Camera control in computer graphics can be classified into three different categories. *Automatic* approach uses the declarative properties (such as its orientation or distance to the character) and path properties for camera. *Reactive* approaches apply and extend notions used in autonomous robotics and sensor planning and the behavior of the camera is driven in direct response to properties in the current image. *Interactive* control systems provide the user with a view of a model world and modify the camera set-up in response to directions from the user. This approach provides a correlation between user input (through mouse, keyboard, and interaction with multi-touch surface or other devices) and the camera parameters. Interactive camera systems are partially automated and allow users to directly change the view. A survey of various camera control techniques and solving mechanisms is described by M. Christie et al [18]. In this work we are mainly interested in interactive camera control approaches.

2.4 Interactive camera control

Interactive camera control system considers user's visualization preferences and provides the user with a view of model world and modifies the position and orientation the camera in response to directions from the user. It maps the dimensions of the input to the camera properties. Therefore it naturally raise the question how user inputs can be mapped into corresponding camera parameters.

2.4.1 Taxonomy of Interaction metaphors

Various interaction metaphors have been defined and implemented by [23, 25] corresponding to the mapping between user input and camera properties. We

prepare a taxonomy of interaction metaphors based on their usefulness in various tasks in virtual environments. Our taxonomy is inspired from [5]. The tasks in virtual environments can be classified as:

- Navigation in virtual environments.
- Object selection
- Object manipulations

The camera control metaphors can be described according to following taxonomy: *direct camera control(D)* metaphor, *indirect camera control(I)* metaphor, metaphor for *object selection(OS)*, and metaphors for *object manipulations(OM)*. Most object manipulation techniques can also be used for object selection tasks. Direct camera control metaphors can be further classified as *object centric(OCD)*, and *user centric(UCD)*. Object centered metaphors allow the user to easily explore a single object, while user centric metaphors are more suitable for scene exploration

- **Eyeball in hand:** in this metaphor, the movements of the input are directly coupled to the virtual camera in an absolute manner, as if the user is holding his ‘eyeball in his hand’. The metaphor requires that the user imagine a model of the virtual environment somewhere in the scene. The eyeball (a camera) is placed at the desired viewpoint and the scene from this viewpoint is displayed. It is a direct camera control metaphor with is suitable for the navigation and can have up to 6DOFs. But the limited workspace for hand limits the scope of the navigation which is true for all absolute user centric metaphors. It is not suited for object selection and manipulation tasks in virtual environments.
- **Scene in hand:** The scene in hand metaphor is almost the opposite of the eyeball in hand metaphor. It is an object centric, direct camera control metaphor. In this metaphor, Instead of moving his or her viewpoint, the user imagines moving the object. In this metaphor, the scene moves in correspondence with the user’s input device. If the input is twisted clockwise the scene rotates clock -wise, if it is translated left the scene translates left, etc. Translations and rotations of the control device result in corresponding translations and rotations of the scene. This metaphor works well with single and reasonably small objects exploring complex environments and landscapes does not seem to be natural. There also exists a problem selecting the center of rotation. Especially when moving through an interior the metaphor clashes with the user’s perception of being enclosed and the linkage of scene motion to hand motion is incongruous and difficult to grasp.
- **Flying vehicle control:** This metaphor is a user centric, direct camera control metaphor which represents the virtual camera as mounted on a virtual vehicle. The movement of the input device controls the position of

the virtual vehicle. This metaphor is widely used solution when the user has to move around in a limited-sized world. This approach is, perhaps, the most prevalent approach for navigating through virtual environments. The main problem lies in avoiding the lost in space problem encountered when the user has multiple of DOFs to manage in either highly cluttered environments or in an open space with a few visual landmarks.

- **UnCam:** This is a camera manipulation metaphor that relies on 2D gestures with a single button stylus or a mouse to directly invoke specific camera operations within a single 3D view. No 2D widgets or keyboard modifiers are necessary. This metaphor is also suitable to use with multi-touch surfaces for camera control[25].
- **Ray-casting and cone-casting:** these are by far the most popular distant selection metaphors. Attached to the user's virtual pointer there is a virtual ray or a small cone. The closest object that intersects with this ray or cone becomes selected. This metaphor allows the user to easily select objects at a distance, just by pointing at them[5].
- **The virtual hand:** This metaphor is the most common 'direct manipulation' technique for selecting and manipulating objects. A virtual representation of the user's hand or input device is shown in the 3D scene. When the virtual representation intersects with an object, the object becomes selected. Once selected, the movements of the virtual hand are directly applied to the object in order to move, rotate or deform it[5].

Table 2.1 presents the taxonomy of interaction based camera control metaphors. Most of these interactive camera control approaches have focused on techniques for *directly* controlling the camera position and orientations.

In our view, this is the source of much of the difficulty. Direct control of the *6DOFs* of the camera (or *7DOFs*, if field of view is included) is often problematic and forces the users to attend to the interface in addition to — or instead of — the goals and constraints of the task at hand. Also, these interaction metaphors can present difficulties when manipulating the virtual camera and/or manipulating 3D objects in virtual environment. In order to achieve smooth cinematographic functionalities with in virtual environments, these low-level, direct controls, must be abstracted into higher level camera primitives, and in turn, combined into even higher level interfaces. This technique has already been successfully applied to interactions within a virtual environment [19]. Also, it is clear that we need to overcome several problems in order to provide an intuitive metaphor for 3D navigation. First of all, standard 2D input devices are not always preferable to control all degrees of freedom. It is also known that disorientation of the user will occur more easily when providing more degrees of freedom.

In the following sections we explore various interactive camera control techniques for various tasks in virtual environments such as exploration for 3D objects, exploration of complex environments.

Metaphor	Full 6 DOFs	UCD	OCD	IC	OS	OM	Applications
Eye-ball in hand	yes	yes	no	no	no	no	navigation
Scene in hand	no	no	yes	no	no	no	navigation
Flying vehicle control	yes	no	no	no	no	no	navigation
UniCam	no	yes	no	no	no	no	navigation
Ray -cast	no	no	no	no	yes	no	selection
cone-cast	no	no	no	no	yes	no	selection
Virtual hand	yes	yes	no	no	yes	yes	selection/manipulation

Table 2.1: taxonomy of interaction based camera control metaphors

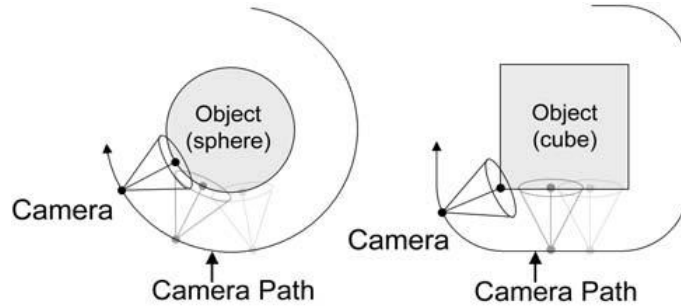


Figure 2.2: HoverCam motion over a sphere and a cube[13]

2.4.2 Exploration of 3D objects

Many applications require close inspection of 3D objects. In order to that several metaphors have been proposed. Ware and Osborne[23] have proposed a simple metaphor called *scene in hand* metaphor. It provides the mapping between movement of an object and input device. This technique shows the benefit when exploring the object as it is held in user's hand. This metaphor allows the user to orbit around objects but it is less efficient for navigation in global environment. Khan et al[13] have proposed an interaction technique called HoverCam for navigating around objects at close proximity while maintaining a fixed distance from the surface and keeping the object roughly centered in the field of view (Fig 2.2). It is based on a "closest point search" algorithm, which determines the next look-at point of the camera. The camera can moves along the surface of the model, and smoothly rotates around its edges while avoiding collision and occlusion with scene objects and local environment

HoverCam algorithm essentially handles all static 3D models whereas moving objects may not always be handled properly, especially if moving faster than the camera. As it uses closest search point algorithm which is time consuming. Consequently, it is inappropriate for low-end computers, where the computational power is limited, such as mobile devices.

Fabrice et al[7] have proposed another 3D camera manipulation technique called ScrutiCam. It is a click-and-drag based new technique for exploration of objects in 3D to perform different camera movements such as zooming, panning and rotating around a model. It is based on target selection. The user selects his or her point of interest and moves it towards the center of the screen in order to align the camera viewpoint with the normal vector (N) of the corresponding area. ScrutiCam uses the visible surface of the model in order to compute camera motion path (Fig. 2.3).

ScrutiCam requires the user to select a point on the screen plane. It uses image based approach to compute N and the resulting target view. The orientation (O) of the target view equals to the opposite of N . The camera position (P) of the computed view is set along the line defined by the target point and

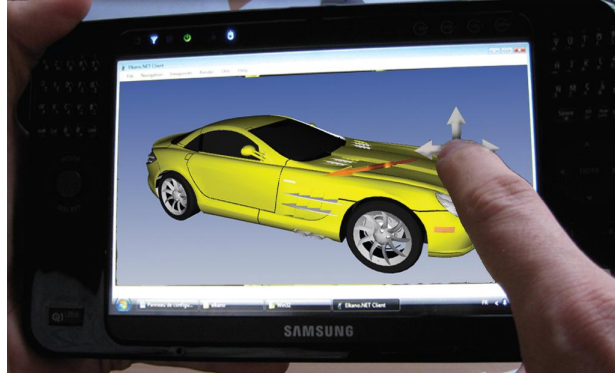


Figure 2.3: Movements to the center of the screen aligns the view[7]

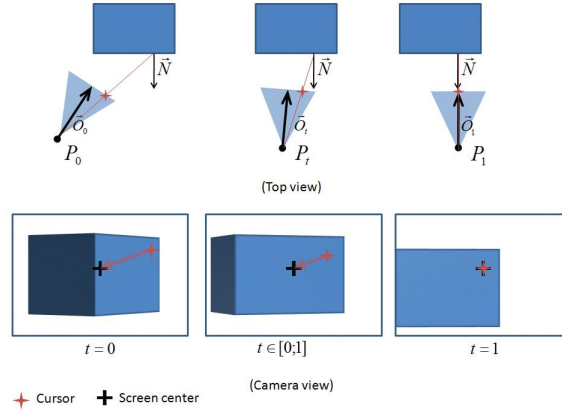


Figure 2.4: Interpolation of camera position P and orientation O to the target area[7].

the normal vector. The distance from the surface of the model to the target camera position can be defined by the user before doing the “click and drag” move, by clicking once on any point of the 3D scene. The distance from the current camera location to the selected point defines the distance that will be used to compute the target camera position (Fig. 2.4). Once the target view computed, a path from the current camera to the target camera is computed as a linear interpolation of the camera position from the current camera position (P_0) to the target camera position (P_1).

Let t be the percentage of advancement of the camera along the path then the next camera position can be calculated as

$$P(t) = P_0 + t * (P_1 - P_0) ; t \in [0, 1].$$

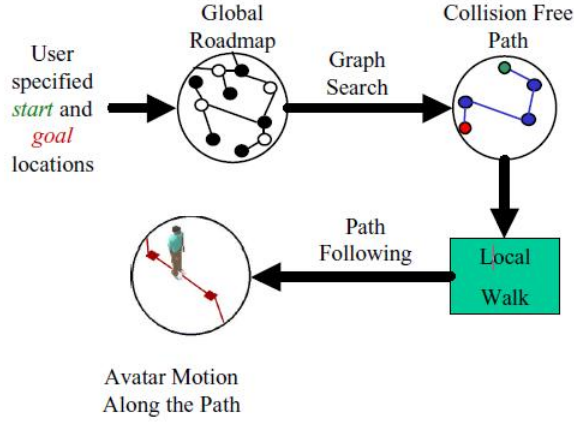


Figure 2.5: An avatar navigating the path at an interactive rate, without colliding with obstacles[22]

This is based on only 2D inputs therefore it can be used with touch surfaces for inspection tasks in virtual environment but it is not suitable for navigation in 3D Virtual environment. A leak point of this technique is that if the surface of the model is very irregular, the local normal vector may change often. As a result, some surprising camera movements may occur.

2.4.3 Exploration of complex environment

Applications of camera control to the exploration of complex environments requires specific approaches that are highly related to the more general problem of path-planning while still ensuring the continuity, smoothness and occlusion criteria. Applications are found both in navigation (looking for a precise target) and in exploration (gathering knowledge in the scene). An interactive navigation in large environments using path planning has been proposed by Salomon et al [22]. It performs graph searching and automatically computes a collision-free and constrained path between two user specified locations. It randomly generates valid configurations in the virtual environment and links them to form a graph. Then it calculates the connectivity of the portions of the model that are accessible to the user’s avatar. When a user is interacting with environment, the resulting roadmap is quickly searched for a path that can take the avatar between the specified start and goal configurations (Fig. 2.5). Because the roadmap graph is fairly sparse and the avatar follows the path in linear segments, paths may sometimes look unnatural, especially when compared to a hand-selected or user-steered path.

Xiao et al[24] has proposed an occlusion free navigation in virtual environment guided by force field. A repulsive force field is considered around each

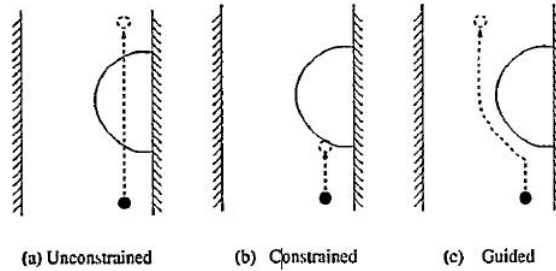


Figure 2.6: Three types of walk-through[24]

object. Force field extends to a pre-determined distance also called region of influence, beyond which no force is applied. If the user is close enough to an object to be inside its region of influence, the object creates a repulsive force to resist any attempt of the user to get closer to the object in question. In this way, the force field around an object forms a barrier to prevent the user from getting too close to the surface of the object and also assists the user to make effective movement in order to avoid the object (Fig. 2.6).

2.4.4 Trough the lens camera control

Gleicher and Witkin [8] have proposed *Through The Lens Camera Control* technique to provide a general solution to the interactive virtual camera control problem by precise control of movements of the object in the scene. Instead of controlling the camera parameters directly, user controls 2D points on the display screen. When the user select some 2D point and move them into new positions, new camera parameters are calculated using constrained optimization, so that the corresponding 3D points are projected into the new 2D points. The required changes of parameters are automatically generated so that the picked screen points are moved in the way the user has specified on the screen.

In Fig. 2.7, the virtual camera is in the position Eye1 and the given 3D point is projected into the 2D point A in the viewing plane. When the user moves the projected point A into a new position at B , the camera parameters are automatically changed so that the given 3D point is now projected into the new position B with new virtual camera position EYE2. The relationship between the velocity (\dot{p}) of m displaced points p on the screen and the velocity(\dot{q}) of the camera parameters q can be expressed through the Jacobian matrix (J) that represents the perspective transformation

$$\dot{p} = J\dot{q}$$

Gleicher and Witkin[?] propose to solve the non-linear optimization problem which minimizes a quadratic energy function that represents a minimal change in the camera parameters

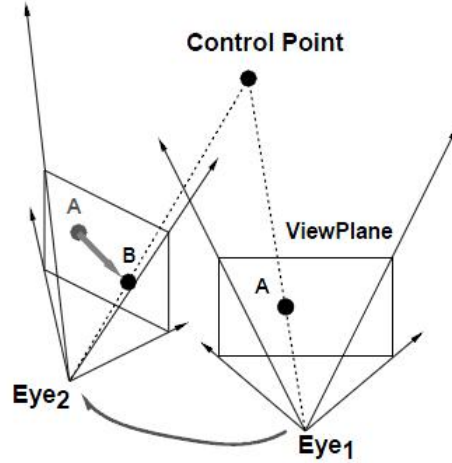


Figure 2.7: Through the lens camera control[8]

$E = 1/2 (q - \dot{q}_0) (q - \dot{q}_0)$ subjected to $(p - \dot{p}_0) = 0$ where (\dot{q}_0) representing the values of the camera's previous velocity. This problem can be converted into a Lagrange equation and solved for the value of λ

$$\frac{dE}{dq} = q, q = J^T \lambda$$

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

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

A simple Euler integration allows us to approximate the next location of the camera from the velocity (\dot{q}_0) is given as

$$q(t + \nabla t) = q(t) + \nabla t * \dot{q}(t)$$

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. However, given m image control points, the Jacobian matrix is derived as a quite complex $2m \times 8$ matrix. However, the Jacobian matrix always has at least one redundant column since its rank can be 7 at most. For the over constrained case of $m \geq 4$, the Lagrange equation is always singular since its $2m \times 2m$ square matrix has rank 7 at most. All these complications result from removing the constraint $q_w^2 + q_x^2 + q_y^2 + q_z^2 = 1$ for unit quaternion $(q_w, q_x, q_y, q_z) \in S^3$ which represent the camera rotations. This approach has the $O(m^3)$ complexity.

An improvement over through the lens camera control technique has been proposed by Kyung et al[16]. It derives a simple $2m \times 7$ Jacobian matrix using an efficient weighted least square method while employing singular valued decomposition. This approach requires only linear time complexity $O(m)$. Also an efficient method for keyframe interpolation has been proposed by Hawkins and Grimm [12]. Instead of interpolation each camera parameter separately, it performs linear interpolation on the camera matrix itself defined as the composition of four matrices: perspective, scale, rotation and translation.

2.5 Conclusion

We have discussed a number of interactive metaphors proposed for mapping user inputs to camera parameters[13, 19, 23]. Each of them has advantages and limitations too. But still there are many aspects that have not been explored yet been explored. To implement a camera control system that is informed by cinematographic principles, number of issues must be resolved. First of all manipulating a virtual camera is generally a delicate task; users cannot deal simultaneously with all seven degrees of freedom. Numbers of interactive mapping metaphors have been proposed[7, 13, 23] but solutions are specific to a given task or set of tasks that limit any generalization. A good metaphor must match the task and it must fit the user's previous knowledge in order to be able to establish a transfer of the user's internal model. Also, the standard 2D input devices are not always preferable to control all DOFs. Occlusion ranging from complete, partial to unoccluded, is nearly always a major issue in camera control. An efficient technique for must be provided to handle the occlusion and collision considering user's priorities of objects in virtual environment as well as the issues related to the performance of that approach. As many 3D applications rely on only a subset of cinematographic techniques to provide the user with an appropriate view in the virtual world, it must be extended in order to provide users an efficient and interactive camera control in virtual environment.

Chapter 3

Tabletop Interaction

3.1 Introduction

The term tabletop refers to the flat surface of a table. This term has been extended computer science as tabletops that allow interaction with data presented on a table. A series of developments have pushed tabletop technology especially multi-touch surface technology to the forefront in last few years, including iPhones, PDAs and mobile phones etc. In this review we present various touch technologies and interaction mechanisms for multi-touch surface and various interactive camera control techniques for multi-touch surfaces

3.2 Touch Technologies

In the last few years, multi-touch surface interaction has inspired researchers all over the world. While the technology required was placed “over the tabletop” in the early systems, recent research focuses on the integration of all components “under the tabletop” or even “in the tabletop”. The following overview presents some important milestones on the way to realizing intuitive multi-touch systems.

3.2.1 Capacitance Based Touch Surfaces

The main advantage of capacitance based systems is their high clarity. They can be operated by fingers or conductive devices such as a wired system.

DiamondTouch

DiamondTouch[6] is a capacitance based multi-user surface. The table surface is constructed with a set of embedded antennas (Fig. 3.1). When a user touches the table, a capacitive circuit is completed from the transmitter, through the touch point on the table surface, through the user to the user’s receiver and back to the transmitter. The system has the ability to identify which users are

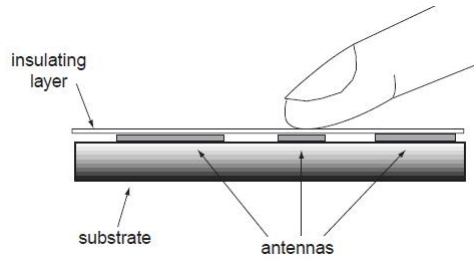


Figure 3.1: Capacitance Based Touch Surfaces[6]

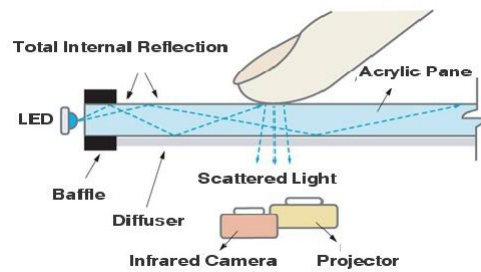


Figure 3.2: Tabletop with FTIR technology[9, 1]

interacting with the surface. It uses a front-projection above the tabletop and has a tracking system that is integrated into the tabletop.

3.2.2 Optical Based Touch Surfaces

Optical based-approaches use image processing and filtering to determine the location of touch position. These technologies use Infra-Red (IR) illumination, and are potentially low cost and scalable.

3.2.2.1 Frustrated Total Internal Reflection (FTIR)

FTIR [9] is a multi-touch IR based technology. It uses a back-projection onto the tabletop's surface and an IR-based tracking system, which also requires a camera underneath the tabletop. At the position where a finger is pressed onto the tabletop's surface, the total internal reflection is distorted and some IR light is coupled out, which can be captured by a camera, which is applied with a special IR pass-through filter. Each touch in FTIR appears as an independent event therefore the FTIR can be used by multiple users but it is impossible to distinguish between them (Fig. 3.2).

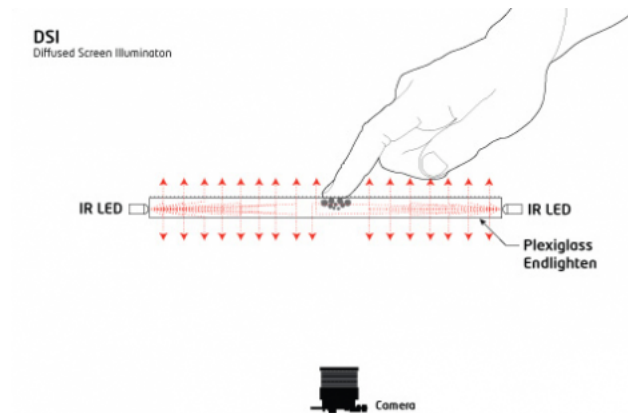


Figure 3.3: Diffused Surface Illumination[1]

3.2.2.2 Diffused Surface Illumination (DSI)

DSI technology is similar to FTIR but it uses an even distribution of IR light over entire surface except at the touch point. It has potentially more problems with ambient IR because of less contrast. It also uses the back-projection and camera under the tabletop (Fig. 3.3).

3.2.2.3 Diffused illumination (DI)

Diffusion Illumination (DI) systems uses the similar hardware as FTIR except that the infrared lighting is placed behind the projection surface. When a user touches the surface it reflects more light than the diffuser or objects in the background; the camera senses this extra light. It uses back-projection and tracking system under the tabletop (Fig. 3.4). DI allows tracking and identification of objects as well as fingers.

3.2.3 Resistance Based Touch Surfaces

Resistive touch screens are usually developed with two layers of electrically conductive material that are separated by a thin layer of space. An electrical current is sent through the layers, and input is registered when the two screens touch. These can detect simultaneous touches with fingers, nails or stylus. It is impossible to have user identification but easy multi-touch and nice screen (LCD) output are the main features of these technologies [1].

3.3 Taxonomy of multi-touch surface technologies

We have provided an overview of the existing multi-touch surface technologies. We now define our taxonomy of multi-touch surfaces based on Interaction prop-

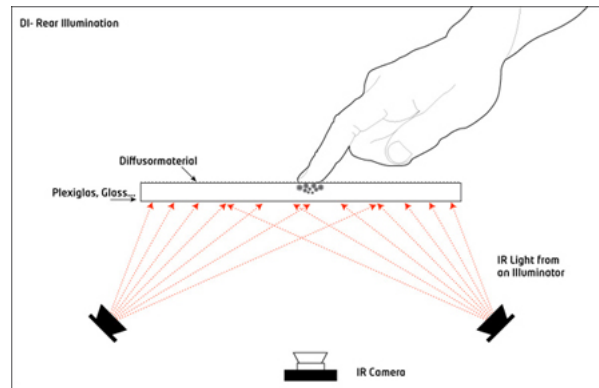


Figure 3.4: Diffuse Illumination system[1]

erties (hand based or device based such as mouse, stylus, phicons etc), tracking and identification based properties (capacitive, resistive, IR-based etc.), image display properties (front-projection, back-projection) and the position of the devices like camera and projector (positions - in, under, or out). Our taxonomy is inspired by [15]. The taxonomy of multi-touch surfaces is presented in table 3.1.

3.4 Interaction mechanism and DOFs for multi-touch surfaces

Various table top interaction mechanisms have been proposed in the literature [7][9]. The study shows that the DOF for interactions tabletops vary according to their types. The desire to manipulate digital data with more DOF using multi-touch surfaces has prompted researchers to develop new interaction techniques.

3.4.1 One-finger for rotation and translation

The one-finger interaction is the simplest mean for control. Single touch provides 2DOF. All the rotation and translation operation can be carried out using a single finger on surface.

3.4.1.1 Rotate and translate (RNT)

It is a tabletop interaction technique that combines rotation and translation into a single gesture [14]. It uses two DOFs. This technique was originally designed for desktop applications but it has been extended to tabletop systems. RNT technique divides the image or object into two regions. When user clicks on image, the RNT detects the position of the control point (inside or outside

Technology	Interaction	Tracking and detection	Image display	In	over	under	user
Capacitive	HB	Integrated into table	FP	T	projector		Multi-user
FTIR	HB	IR-Based Blob detection	BP			Proj, Cam	Single
DSI	HB/DB	IR illumination techniques	BP			Proj, Cam	Single
DI	HB/ body	IR filter	BP			Proj, Cam	Single
Resistive	HB DB	Resistive (electrically)	LCD	T, D			Single

Table 3.1: Taxonomy of multi-touch surface technologies

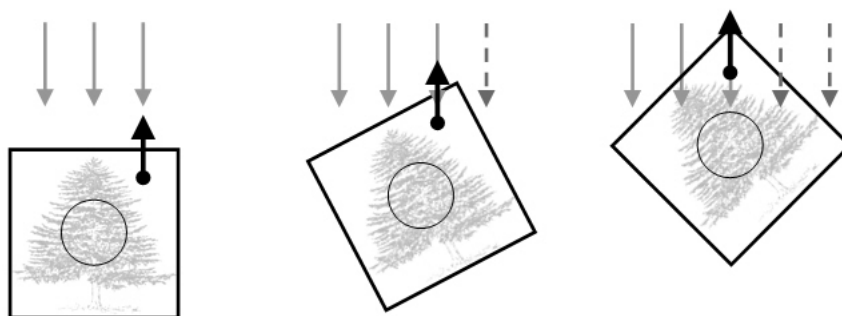


Figure 3.5: Unbalanced movement resulting in upward translation and counterclockwise rotation from a control point located in the upper-right corner of the object[14]

the circle) (Fig. 3.5). When user click inside the circular region, RNT algorithm performs translation operating following the pointer movement. When user click outside the circular region, RNT algorithm detects the position of the control point and performs both rotation and translation simultaneously following the pointer movement.

3.4.1.2 TNT technique

An improved rotation and translation technique called TNT has been proposed in[17]. With this method, a person can place their hand or a physical block on a virtual object and the position and orientation of the finger or block controls the movement and rotation of the virtual object.

3.4.2 Two fingers for zoom-rotation-translation

On multi-touch tables, two fingers are typically used for a combined movement, rotation and scaling of a virtual object. The position of the first touch is used to determine the movement of the object and the position of the second touch relative to the first is used to determine the rotation and scale.

3.4.2.1 Sticky tool

Hancock et al[11] has presented an interaction paradigm called *sticky tool* for 3D environment with full 6DOFs manipulation. It is composed with sticky fingers (a technique for moving, rotating, and lifting virtual objects), opposable thumbs (a method for flipping objects over). Because ones fingers stay in touch with the virtual object in the location they initially contact, this can be referred to as a *sticky-finger* interaction. Sticky finger provides move (in x and y), spin (rotate about z) and scale. As the finger moves, the virtual object moves with

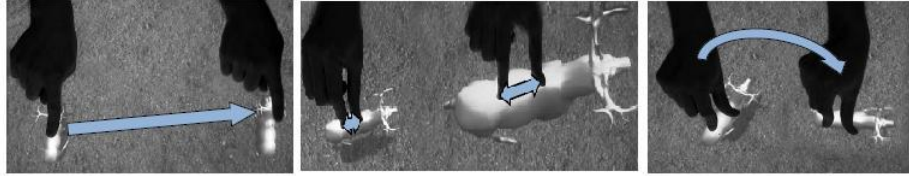


Figure 3.6: Translation, Lift and 2D rotate operation using Sticky finger[11]

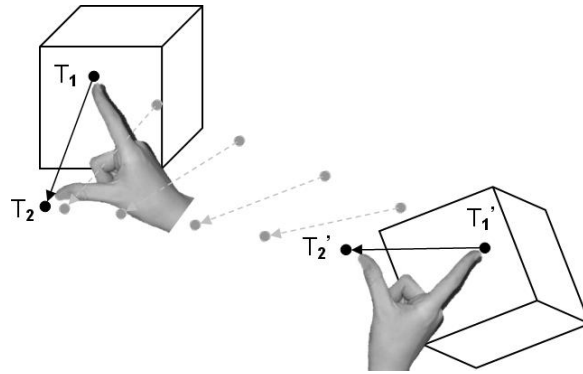


Figure 3.7: Simultaneous rotation and translation using shallow depth technique[10]

them and as the fingers rotate relative to one another, so do the virtual objects (Fig. 3.6).

3.4.3 Three Fingers interaction

The three-touch allows the most efficient means for control because users can concurrently and independently manipulate all six DOFs (3DOF for translation and 3DOF for orientation). Various techniques have been proposed for three finger interactions on multi-touch surface.

3.4.3.1 Shallow-depth 3D interaction

Traditional 2D control technique RNT [14] has been extended by Hancock et al[10] to allow shallow-depth 3D interaction, restricted to the same level of depth, using three-touch interaction. This technique maps 6DOFs input to 5 or 6DOFs output. In this mapping, the first point of contact is used for translation, the second point for yaw about the first point, and the third point for pitch and roll about the centre of the object. The depth can be specified by the difference in distance between the first and second touch points (Fig. 3.7).

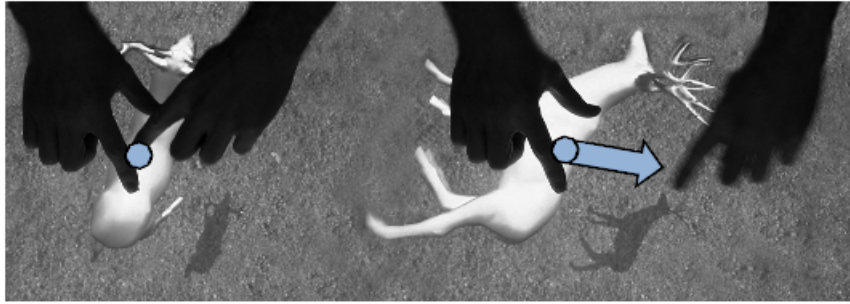


Figure 3.8: Rotate 3D Object using Sticky fingers and opposable thumb[11]

3.4.3.2 Sticky tool

Sticky tool [11] also uses the third finger (opposable thumb) along with two sticky fingers to flip the object about x and y axis. It maps first two fingers to move (in x, y, and z) and rotate (about z). The third finger is then used for rotation about x and y. This technique provides full control of all 6DOF, enabling behavior such as lifting objects and flipping them over (Fig 3.8). This technique is suitable for multi-touch surfaces. This technique mainly focuses on selection and manipulation of the virtual objects.

Table 3.2 presents the summary of various interaction techniques.

3.5 Multi-touch surface Interaction and Camera control

Camera manipulation is an important problem in computer graphics (e.g. games, simulations etc). Navigation in a 3D environment and inspecting 3D objects are some common 3D tasks. Many 3D applications use the classical controls dedicated to camera manipulations that perform camera movements such as translation, rotation and zoom. Camera control in virtual environments requires at least 6 DOF for output. These DOF can be directly controlled by the inputs using various metaphors. Many techniques such as Navidget[19] and ScrutiCam[7] use user's gestures on multi-touch surface to control a camera.

3.5.1 Navidget

Navidget[19] provides an interactive technique that provides control for fast and easy camera positioning in the 3D environment from 2D inputs. It is based on point of interest (POI) technique. It lets the user control where they want to look, with simple 2D gestures like push, move and release. It uses a 3D widget composed of a half-sphere facing the current viewpoint, a border ring, a set of size actuators, and a virtual camera, in order to let the users control the viewing direction at destination (Fig. 3.9). By moving a 3D cursor on this surface, users control how they want to visualize the target. This technique is well suited for navigating in large environment, but this can be harder to

Interaction technique	DOFs for input	Operations
RNT	2	Rotation (z), translation (x, y)
TNT	3	Rotation (z), translation (x, y)
Shallow Depth	6	Rotation (z), translation (x, y), flipping (rotation x and y)
Sticky Tools	6	Rotation (z), translation (x, y), flipping, (rotation x and y), lifting

Table 3.2: Interaction techniques and their properties

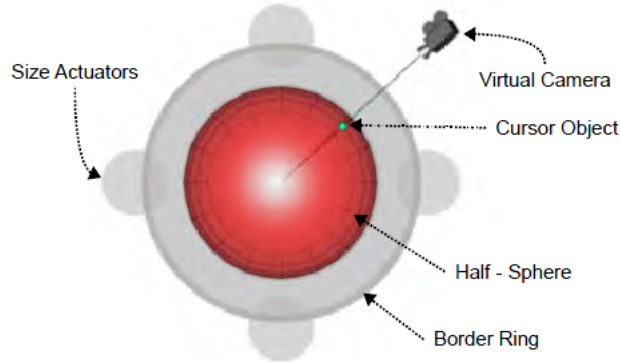


Figure 3.9: The Navidget[19]

use for precise camera movements, as required when inspecting details in close views.

3.5.2 SrutiCam

ScrutiCam[7] is also a 3D camera manipulation technique that is based on POI technique and also inspired by the “Traceball Technique”. ScrutiCam can be easily controlled, as it only uses two gestures: moving to the center of the screen to smoothly align the view to the surface of the model, and moving in any other direction to translate the view along the screen plane (Fig. 2.3). It needs only 2D inputs therefore it can be used with touch surfaces for inspection tasks in virtual environment but is not suitable for navigation in a 3D virtual environment.

3.5.3 Follow My Finger navigations(FmF)

Ajaj et al[4] has proposed FmF technique for navigation in 3D virtual environment using a 2D interactive view on multi-touch surfaces. At 2D level, FmF navigation technique is similar to RNT[14]. FmF uses the 2D interactive horizontal view of a virtual scene projected on a multi-touch surface and a 3D view of the same scene presented vertically. It uses a 2D camera icon linked to a 3D point of view for 3D manipulation. This technique controls both 2D rotation and translation transformations of the 3D view point.

The literature study[4, 7, 10] shows that many different techniques have been proposed for camera control using interactive tabletops, but controlling a camera interactively in a 3D environment remains a difficult task, and novel interaction techniques still need to be designed.

3.6 Using multiple fingers for 3D interaction

Multi-touch surfaces have high potential for 3D interaction, including through the large number of degrees of freedom that can be monitored simultaneously during the use of several fingers. Kruger et al [14] proposed RNT technique for translation and rotation using one finger which provides up to 3 DOFs. Hancock et al [10] implemented *shallow depth* technique using one, two and 3 fingers for controlling translation, rotation and limited z-depth and provides up to 6 DOFs. Sticky tool [11] use 2 sticky fingers along with one thumb for interaction and provides up to 6DOFs. Martinet et al[20] proposed Z-technique for 3D positioning of objects using 3 fingers.

The control of multiple degrees of freedom has traditionally been the main issue of research interaction in 3D. Indeed, the most basic transformation of the interaction 3D can be achieved along several axes, and totally free navigation in 3D environment requires control of all these degrees of freedom simultaneously. Also, the use of cinematographic techniques for interactive camera control on multi-touch surfaces requires to identify and associate numbers of gestures with various combinations of finger movement on surface.

The use of a conventional touch surface providing only two degrees of freedom, it is necessary to use techniques to easily choose the degrees of freedom to work on. However, the screens allow multi-touch control to consider a larger number of degrees of freedom simultaneously. Indeed, on multi-touch surfaces, it is possible to interact with several fingers. As each finger can move in several directions, each of them can potentially control multiple DOFs. However, the mobility of different fingers is not complete and must be taken considerations anatomical limitations of the hand to propose a control that is comfortable and easy to perform.

3.7 Technical Challenges

Each device has advantages and disadvantages. The main drawback of multi-touch surfaces is the imprecision. The users typically touch the interaction surface at multiple points with their hands and not only at the intended interaction point of the device. So, the system should be able to distinguish between intended and unintended input in order to generate correct results. This situation becomes even more complex if users consciously use their fingers in addition to devices to interact with the system. Users of tabletop system do not share a common perspective for the display of information in shared environment. Also, the horizontal surface offer a poor surface for perspective effect. One of the considerable points is that the 3D contents have been mainly studied for object manipulation and not for navigation.

3.8 Conclusion

In summary, the tabletops can be used to interact with 3D virtual environment. However, their use for interactive camera control in 3D virtual environment is difficult because of lack of DOFs and unavailability of appropriate techniques.

It can be interesting to use cinematographic techniques for interactive camera control on multi-touch surfaces. Also interaction with a 3D scene is a complex task because it seeks to control many parameters simultaneously. It will be important take into account the difficulty of learning gestures, difficulty in use and the performance of use.

In the following chapters, we present an interaction metaphor so called *frame metaphor*, to interactively control the camera using multi-touch. It allows the user to navigate in 3D environment by controlling the frame using natural hand gestures. We then present results and comparisons with other techniques.

Chapter 4

Our Approach: Tabletop Interactive Camera Control

4.1 Introduction

In cinematography, a cinematographer uses a physical frame to appropriately view and compose elements in the scene. We intend to extend this idea to interactively control a camera on a multi-touch surface by controlling the viewing frame. We propose an interaction-based frame metaphor for camera control on multi-touch surfaces to that allows the user to freely navigate in 3D environment using natural hand gestures. We define a set of mappings between user inputs and camera parameters. In the following sections we present our approach to interactive camera control using this frame metaphor followed by design architecture and show how we propose to build upon the *through the lens camera control* approach [8, 16] and direct approach and how to map finger gestures to 3D movements.

4.2 Frame metaphor

Frame metaphor is a high level interaction metaphor in which the position and speed of the camera is indirectly controlled by manipulating a virtual frame. We propose a *tabletop interactive camera control* scheme to provide a solution for virtual camera control problem based on a subset of cinematographic properties (viewing angle, viewing distance and occlusion avoidance). Instead of controlling the camera parameters directly, user controls an image frame displayed in the center of the multi-touch surface representing the current view of the camera. Most of the metaphors proposed in literature [5, 13, 23] are based on direct mapping between inputs and parameters of the virtual camera. Our frame metaphor provides an indirect control of camera parameters. Users can perform rotation, translation, zooming, tilting and many other cinematographic operations and can deal simultaneously with all seven DOFs (3DOFs for translation, 3DOFs for orientation, and 1DOF for scaling) in highly dy-

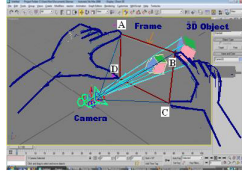


Figure 4.1: Interactive camera control with frame ABCD

dynamic environments by manipulating the frame through natural hand gestures on multi-touch surfaces. The approach intends to offer a natural control of what we see in the environment and how we move the camera. This metaphor proposes a direct control what we see in the environment.

Frame-based metaphor has some similarities with the taxonomy O3/V1 and O3/V2, in which they use two copies of the explored synthetic world and thus two windows, one in each of these worlds. The window in the primary world, through which the user views the secondary world, called output window (Wo). The virtual counterpart of this window in the secondary world we call viewing window (Wv). They use a “pen” camera widget to define the secondary viewpoint in the surrounding virtual environment and allow users to navigate and remotely modify the objects in 3D world [ref]. In contrast to this, frame metaphor allows users to control the position and orientation of the camera by manipulating the rectangular frame. A rectangular frame (ABCD) displayed in the center of the interactive multi-touch surface represents the current view of the camera. This frame, whose four vertices can be manipulated and interactively deformed by the user for each type of deformation (translation, rotation, trapezoid, etc.), is associated with a primitive cinematographic motion (pan, zoom - in / out, arcing, etc.). The composition of these strains allows the composition of camera movements, and therefore provides a control of the camera through its image.

4.3 Tabletop Interactive Camera Control (TICC) System

The central notion of TICC is that camera placement and movement is usually done for particular operations and we can identify these operations based on analysis of the tasks required in the specific job at hand. By analyzing a wide enough variety of tasks, a large base of primitives can be easily drawn upon to be incorporated into a particular task-specific interface.

TICC system mainly consists of two main modules:

- Filter module
- Mapping module

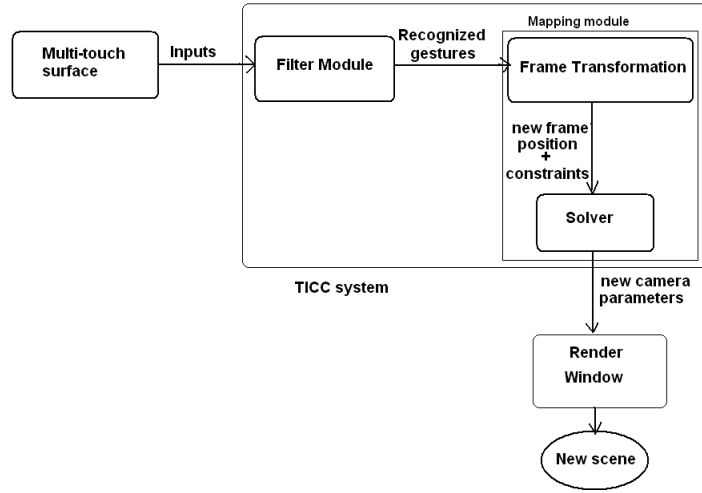


Figure 4.2: TICC system architecture

4.3.1 Filter Module

The main tasks of this module are:

- Filter the inputs received from multi-touch surface and store and register these valid inputs for processing.
- Once the inputs are registered, the module associates appropriate gestures with these inputs based on number of inputs, their motions and some other properties of the system such as the mode of operation, previous state of system etc.

Each input corresponds to 2DOFs in the system, therefore using multiple fingers for interactions can lead to achieve more DOFs in the system and take real advantages of multi-touch technologies. A number of interaction techniques have been proposed. As we discussed in chapter 3, Kruger et al[14] proposed RNT technique for translation and rotation using one finger which provides up to 3 DOFs. Hancock et al [10] implemented *shallow depth* technique using one, two and 3 fingers for controlling translation, rotation and limited z-depth and provides up to 6 DOFs. Sticky tools[11]use 2 sticky fingers along with one thumb for interaction and provides up to 6DOFs. We are interested in interactive camera control with cinematographic primitives. Therefore we need to identify and associate various gestures with these inputs. Also these gestures should be natural to use and easy to learn by the users. Using multiple fingers for interaction opens the scope to associate natural gestures with them. Since the frame metaphor uses a rectangular frame through which user can control the position and orientation of the camera, we can use up to 4 fingers to define a number of gestures such as translation, rotation, tilting etc. associated with a

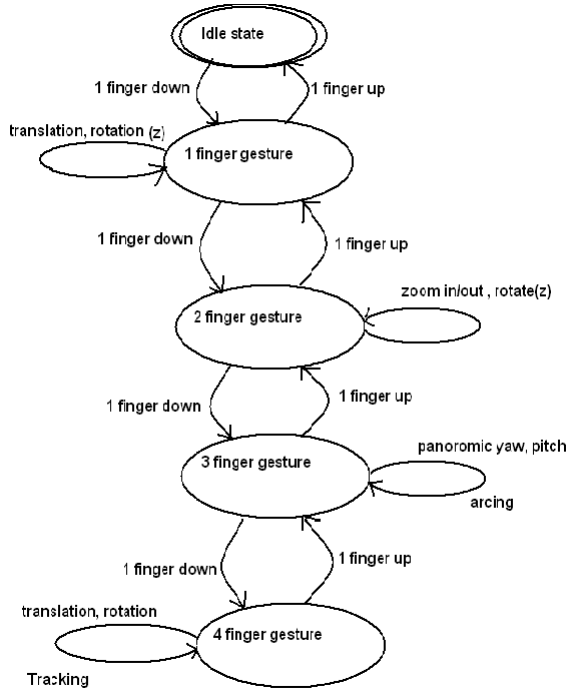


Figure 4.3: State transition diagram for gestures

primitive motion films (pan, zoom - in / out, arcing, etc.). The state transition diagram in Fig. 2 describes the interaction mechanism. It is based on finger down, finger up and finger move events coming from the table SDK. The ids associated to each finger are used to find which finger touched the table first and keep track of each individual finger.

4.3.2 Mapping Module

Mapping module consists of 2 components:

4.3.2.1 Frame transformation

This module performs the 3D transformations on a rectangular frame based on the operation selected by the filter and computes new coordinates of the frame.

4.3.2.2 Solver

In this system, the solver can be viewed simply as a black box that takes frame parameters as input and produces new values for camera parameters. When user moves a frame to new positions, new camera parameters are calculated

Figure 4.4:

using properties of the frame. Instead of controlling the camera parameters directly, we propose an indirect camera control by controlling a 3D rectangular frame in virtual environment with reference to which new camera parameters are calculated. Use of 3D rectangular frame helps the user to establish the relation between mental image of the scene and the view produced by camera.

Gleicher et al [8] proposed through the lens camera control technique which is a powerful technique of camera control. It is a nonlinear optimization technique in which the 2D points displayed on the screen are controlled by the user and the required changes in camera parameters are calculated automatically so that the selected points on the screen move to target locations specified by the user. Kyung et al [16] have improved this technique by considering this problem as constrained inversion problem. It derives a simple $2m \times 7$ Jacobian matrix using an efficient weighted least square method while employing singular valued decomposition. We propose to build the solver for interactive camera control using through the lens camera control [16] approach.

The general construction scheme is as follows: user manipulates the rectangular frame using hand gestures on multi-touch surface. The projection of vertices of new frame position is considered as a source input control for the solver. For simplicity we specify four 2D coordinate points on the display screen as the target for the solver. The solver computes new camera parameters in such a manner so that the projections of rectangular frame coordinates are identical to target scene (Fig. 4.4).

A simple camera model based on Euler angles can be described by 7 parameters: 1 for focal length, 3 for camera position 3 for orientation for the camera. The use of Euler angels suffer from singularities such as gimbal lock. Gimbal lock is the loss of one degree of freedom that occurs when the axes of two of the three gimbals are driven into the same place and cannot compensate for rotations around one axis in three dimensional space[2]. Quaternion representations can be used for representing orientations because they are free from gimbal lock. We define camera parameter vector

$$x = [f, u_x, u_y, u_z, q_w, q_x, q_y, q_z] \in R^3 \chi S^3$$

where $f \in R$ is focal length, $(u_x, u_y, u_z) \in R^3$ represents camera position and $(q_w, q_x, q_y, q_z) \in S^3$ is the unit quaternion for the camera rotation. Given 4 points $(p_1, p_2, p_3, p_4) \in R^3$, the perspective viewing transformation for 4 points $V_p : R^4 \times S^3 \rightarrow R^{2m}$, is defined by

$$V_p(x) = h \dots \dots \dots (1)$$

where $P = (p_1, \dots, p_m) \in R^{3m}$; m is number of control points. and $h = (x_1, y_1, \dots, x_m, y_m) \in R^{2m}$ with each (x_i, y_i) being the perspective viewing transformation of p_i on to the 2D display screen. Through the lens camera control techniques proposed by Gliether et al[8] and Kyung et al[16] focus on computing the camera parameters x for the given a 3D point P and perspective viewing transformation V .

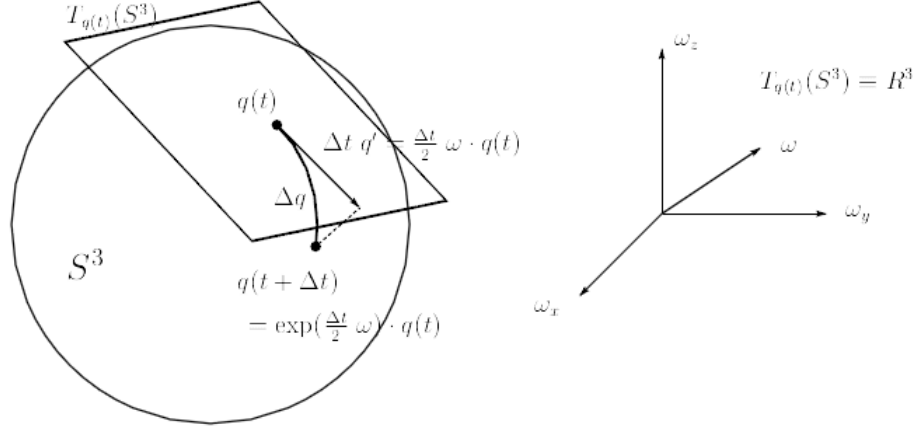


Figure 4.5: computing $q(t + \Delta t)$

Let the position and orientation of the virtual camera at time t be given by $u(t) \in R^3$ and $q(t) \in S^3$. For a given point $p = (x, y, z)$ in the world coordinate system, the projected 2D image point $h(t) \in R^2$ can be represented by

$$h(t) = P_{f(t)} \circ R_{u(t)} \circ T_{u(t)}(P)$$

$$h(t) = V_p(x(t))$$

where $P_{f(t)}$ is the perspective projection with a focal length $f(t)$, $R_{q(t)}$ is the rotation of unit quaternion $q(t)$. The perspective projection of V_p is rational expression that produces the nonlinear relationship between the camera parameters and the projected 2D images.

We use the quaternion rotation matrix proposed by Glicher et al[8] in which they eliminated the unit length constraint on unit quaternion i.e. $(q_w + q_x + q_y + q_z) = 1$.

$$R_{q=\frac{2}{|q|^2}} \begin{bmatrix} \frac{|q|^2}{2} - q_y^2 - q_x^2 & q_x q_y + q_w q_z & q_x q_z - q_w q_y & 0 \\ q_x q_y - q_w q_z & \frac{|q|^2}{2} - q_x^2 - q_z^2 & q_w q_x + q_y q_z & 0 \\ q_w q_y - q_x q_z & q_y q_z - q_w q_x & \frac{|q|^2}{2} - q_x^2 - q_y^2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The key concept is solving for the time derivatives of the camera parameters, given the time derivatives of the controls. User interacts with the rectangular frame using hand gestures on multi-touch surface. The velocity is calculated from the motions of the cursors on multi-touch surface. The use of differential control does not allow user to directly position the camera in virtual environment, but instead it provides a means of translating adjustments of the controls into continuous motions of the camera.

Let $p = (x, y, z) \in R^3$ be a point in 3D space, $q \in S^3$ be an unit quaternion and the point p is considered as a quaternion $(0, x, y, z)$, the 3D rotation $R_q \in SO(3)$ can be defined as

$$R_q(p) = q.p.\bar{q}$$

where \bar{q} denotes the conjugate of q . The Lie group structure of S^3 provides the canonical coordinate system of $R^3 \equiv T_1(S^3)$ to every tangent space $T_q(S^3)$ for $q \in S^3$ [21](Fig. 4.5). The derivative of unit quaternion curve $q(t)$ is given as

$$q'(t) = (t).q(t) \text{ for some } t \in R^3$$

$$\text{which implies to } q'(t) = v(t).q(t)$$

$v(t) \in R^3$ be the velocity of point p . It generates a path

$$\hat{p}(t) = R_{q(t)}(p); \hat{p} \in R^3$$

and the derivative of \hat{p} is given as $\hat{p}' = \omega(t) \times \hat{p}(t)$ where $\omega(t) \in R^3$ is the angular velocity and $\omega(t) = 2v(t)$. When an angular velocity ω is computed, the quaternion $q(t)$ is updated to a new quaternion $q(t + \Delta t)$ by

$$q(t + \Delta t) = \exp\left(\frac{\Delta t}{2}\omega\right).q(t)$$

perspective viewing transformation $h(t)$ can be rewritten as

$$h(t) = P_{f(t)} \circ R_{u(t)} \circ T_{u(t)}(P)$$

$$h(t) = P_{f(t)}(\hat{p}(t)) = P_{f(t)}(\hat{x}(t), \hat{y}(t), \hat{z}(t))$$

$$h(t) = \left(\frac{f(t)\hat{x}(t)}{\hat{z}}, \frac{f(t)\hat{y}(t)}{\hat{z}} \right)$$

Therefore, for each of the 4 control points p_i , we construct a simple 2×7 jacobian matrix [16] and therefore the we have the jacobian matrix of size 8×7 .

The jacobian matrix is not a square matrix but it is a singular matrix.

$$J = \begin{bmatrix} \frac{\hat{x}}{\hat{z}} & \frac{fR_{11}}{\hat{z}} - \frac{fR_{31}\hat{x}}{\hat{z}^2} & \frac{fR_{12}}{\hat{z}} - \frac{fR_{32}\hat{x}}{\hat{z}^2} & \frac{fR_{13}}{\hat{z}} - \frac{fR_{33}\hat{x}}{\hat{z}^2} & \frac{f\hat{x}\hat{y}}{\hat{z}^2} & f + \frac{f\hat{x}^2}{\hat{z}^2} & -\frac{f\hat{y}}{\hat{z}} \\ \frac{\hat{y}}{\hat{z}} & \frac{fR_{21}}{\hat{z}} - \frac{fR_{31}\hat{y}}{\hat{z}^2} & \frac{fR_{22}}{\hat{z}} - \frac{fR_{32}\hat{y}}{\hat{z}^2} & \frac{fR_{23}}{\hat{z}} - \frac{fR_{33}\hat{y}}{\hat{z}^2} & -f - \frac{f\hat{y}^2}{\hat{z}^2} & \frac{f\hat{x}\hat{y}}{\hat{z}^2} & \frac{f\hat{x}}{\hat{z}} \end{bmatrix}$$

We need to compute the camera parameters x so that the given 4 control points $p_i \in R^3$ of the rectangular frame are to projected onto 4 target 2D screen point $h_i \in R^2$.

$$V_{p_i}(x) = h_i$$

The perspective projection of V_p is a rational expression that depicts the nonlinear relationship between the camera parameters x and the projected 2D images h . By integrating the velocity, we obtain the solutions of these nonlinear equations. By differentiating these nonlinear equations we obtain linear equations which are solved using Newton's method. The unknown in these equations are the *velocity of the camera parameters* denoted by $\dot{x} = (f', u'_x, u'_y, u'_z, \omega) \in R^7 \rightarrow R^4 \times T_q(S^3)$.

Let $F(x) = V(x) - h_0$; The solution x for $F(x) = 0$ satisfies $V_p(x) = h_0$. For 4 control points p_i we have

$$F(f, u_x, u_y, u_z, q_w, q_x, q_y, q_z) = \begin{bmatrix} \frac{f(\hat{p}_1)_x}{(\hat{p}_1)_z} - (h_1)_x \\ \frac{f(\hat{p}_1)_y}{(\hat{p}_1)_z} - (h_1)_y \\ \frac{f(\hat{p}_2)_x}{(\hat{p}_2)_z} - (h_2)_x \\ \frac{f(\hat{p}_2)_y}{(\hat{p}_2)_z} - (h_2)_y \\ \frac{f(\hat{p}_3)_x}{(\hat{p}_3)_z} - (h_3)_x \\ \frac{f(\hat{p}_3)_y}{(\hat{p}_3)_z} - (h_3)_y \\ \frac{f(\hat{p}_4)_x}{(\hat{p}_4)_z} - (h_4)_x \\ \frac{f(\hat{p}_4)_y}{(\hat{p}_4)_z} - (h_4)_y \end{bmatrix} \text{ j}$$

The differentiation of $F(x)$ is the jacobian matrix $J(x)$, hence we can obtain following linear equation

$$J(x^{(i)})(\Delta x^{(i)}) = -F(x^{(i)}) \dots \dots \dots (2)$$

Therefore

the next camera parameter vector $x^{(i+1)}$ is obtained by

$$x^{(i+1)} = (f^{(i)} + \Delta f^{(i)}, u^{(i)} + \Delta u^{(i)}, \exp(\frac{\omega^{(i)}}{2}).q^{(i)}) \in R^4 \times S^3$$

The jacobian matrix is not a square matrix, it is singular. Therefore we compute least square solution Δx for equations (2). Since we have 4 control points, the system is always over constrained, we use singular valued decomposition (SVD) for jacobian $J(x)$ and construct pseudo inverse $J(x)^+$. For the case $m \geq 4$ the pseudo inverse requires the SVD of $2m \times 7$ matrix that takes $\mathcal{O}(m)$ computation time.

In this approach of camera control, the control points can be served as constraints, maintaining their values as others are changed. This approach worked well with most of the gestures like translation, zoom-in, zoom-out, rotation around z axis but the results for rotation around x axis are not satisfactory. The derivatives of some of the camera parameters are too high which in turn results in rotating the camera in the wrong direction. We therefore provided an approach to directly control the camera.

In this approach, we consider a 2D rectangular frame displayed on the screen. The user interacts with the frame. The filter recognizes the gestures and manipulates the frame. The changes in frame positions, orientation and the type of gestures applied to the frame directly correspond to the new camera parameters. The approach allows the user to control position and orientation of the camera directly in order to be able to perform various cinematographic operations using multi-touch surfaces.

4.4 Implementation

We implemented TICC system prototype in two versions. In the first version, we build the solver using the concept of indirect camera control approach proposed by [8, 16]. The second version is implemented using direct camera control approach. We used Ogre3D graphics engine with visual studio 2005. For SVD decomposition and other matrix operations we used gnu scientific library (GSL). We use TUIO[3] protocol for interaction between multi-touch surface and the application. TUIO is an open framework that defines a common protocol and API for tangible multi-touch surfaces. The TUIO protocol allows the transmission of an abstract description of interactive surfaces, including touch events and tangible object states. This protocol encodes control data from a tracker application (e.g. based on computer vision) and sends it to any client application that is capable of decoding the protocol. We tested our prototype on STANTUM multi-touch surface



Figure 4.6:

4.4.1 Controls

User can work in two different modes. In frame mode user interacts with the rectangular frame using hand gestures on multi-touch surface and can perform various cinematographic operations by controlling the camera position and orientation. In this mode of working, the user can select using the shallow depth technique[10] for interaction with multi-touch surface or user can choose mixed technique mode which is implemented using the concepts of various interaction methods[9, 10, 11, 14]. When user is in frame mode he/she can not interact with objects in the 3D environments.

In the object mode user can interact with the objects in the scene. User can perform selection of the objects and can manipulate them using hand gestures. In this mode user cannot interact with the camera.

4.4.2 Gesture specifications

User can interact with the system using multiple fingers. Use of more fingers gives more DOFs for interaction. We are mainly interested in controlling the camera to provide cinematographic operations; hence the gestures must be

natural and easy to learn. We associated the gestures with reference to the number of fingers and the direction of their motion.

4.4.2.1 One finger gestures

If the user is in frame mode he can use 1 finger for translation, rotation and object selection. If initial position of the finger is inside radius r from the center of the rectangle, user can rotate the frame along z axis. User can perform simultaneously rotation and translation by initially touching the frame between circle and frame boundary[24]. User can perform translation by touching elsewhere on multi-touch surface. User can also select the object by touching the finger over the object. Selection of targets can be used to perform arcing using 3 fingers (arc is a circular motion of camera around a target) . If user is in object mode he can select and translate the object in virtual environment.

4.4.2.2 Two finger gestures

User can use 2 fingers for zoom and rotation operations in both frame mode and object mode. Zoom in/out can be performed by moving both the fingers in opposite direction. If one finger is constant and the other finger is moving, it results in rotation along z -axis.

4.4.2.3 Three finger gestures

If the two fingers are fixed and the third finger is moving, user can perform arcing if some object is selected otherwise user can perform panoramic yaw and pitch operations.

4.4.2.4 Four finger gestures

User can perform translation operation by moving all the four fingers in same direction. The tracking can be performed by moving first 2 fingers in one direction and the last 2 fingers in opposite direction.

The summary of hand gestures is presented in (Fig. 4.7).

4.4.3 Top View window

This window displays the top view of the scene using another camera that always look at the main camera of the system. Use of top view window helps the user to control the position and orientating of the camera by providing the global view of the virtual environment. Also it helps the user to avoid collision of the camera and occlusion of the scene.

4.5 Results and discussion

An experimental result of TICC system using *through the lens camera control* technique [16] is demonstrated in Fig 4.8, Fig 4.9 and Fig 4.10. The user

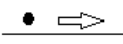
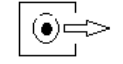
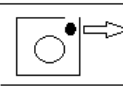

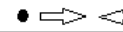
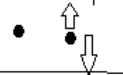


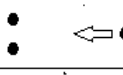
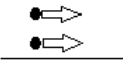

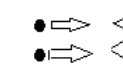
Finger Motion	Gesture in Frame mode	Gesture in Object mode
	Translation	Selection, Translation
	Rotation around z-axis	
	Combined rotation and translation	
	Zoom In	Scale in
	Zoom out	Scale out
	Rotation around z-axis	Rotation around z-axis
	Panoramic yaw	
	Panoramic pitch	
	Arcing around object	
	Translation	
	Tracking	
		

Figure 4.7: Summary of hand gestures for TICC

interacted with rectangular frame and transformed it from current position (Fig 4.8) to new position (Fig 4.9). Table 4.1 presents the values of camera parameters after each of 10 Newton’s interpolation, that brings the projected image of frame back to the initial position (Fig 4.10).

TICC system always has 4 control points, therefore the system is always over-constrained and the Newton’s method may result in approximation errors. This approach worked correctly for translation, zoom in/out and rotation around z axis, but the results for rotation around xaxis were not correct. We spent most of our time in analyzing and finding the cause of this problem. In this case, we found that variations in quaternion parameters in the jacobian matrix were very large that moved the camera in wrong direction. We experimented various solutions but not yet completely succeed to achieve the desired



Figure 4.8: Initial scene



Figure 4.9: User manipulates rectangle using hand gestures

results. We implemented an alternative solutions by directly mapping the user inputs to camera parameters. Both of these approaches has their advantages and limitations. User can more precisely control a virtual camera using indirect approach but it need more computation time than the direct mapping between user inputs and camera parameters.

We compared TICC with ScrutiCam proposed by Fabrice et al[7]. ScrutiCam camera control technique is suitable for inspection tasks and uses only 2D input. It is not suitable for navigation in virtual environment; whereas TICC allows the user to navigate in virtual environment by controlling all 7 DOFs of camera. TICC uses multi-finger gestures and hence is suitable for large screen multi-touch surfaces. One of the leak point in TICC is that it may not be suitable for small multi-touch devices because of the limited and small size of the multi-touch screen. Use of multiple fingers on small multi-touch devices such as PDA, iPhones etc.. may result in occlusion of the scene.

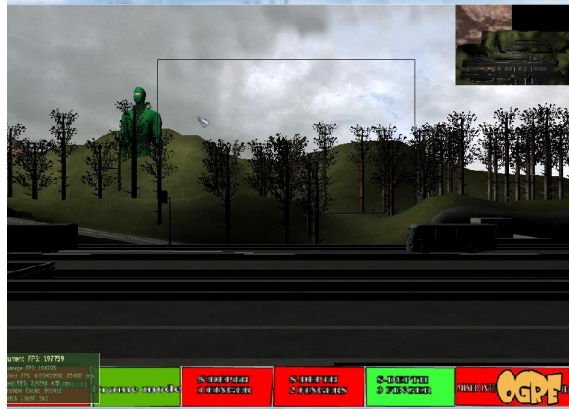


Figure 4.10: Scene viewed from new camera position

frame No	u_x	u_y	u_z	q_w	q_x	q_y	q_z	f
1	0	10	0	0	0	1	0	1
10	12.88	13.51	2.19e-15	-1.28e-17	-4.17e-18	1	1.10	1
20	18.87	15.14	2.35e-15	-1.04e-17	-6.38e-18	1	-1.68e-17	1
30	22.48	16.11	3.77e-15	-9.47e-18	-5.48e-18	1	-6.22e-18	1
40	24.19	16.67	3.30e-15e	1.20e-17	-3.19e-18	1	-4.58e-18	1
50	25.72	17.01	3.02e-15	-1.39e-17	-2.73e-18	1	-3.05e-18	1
60	26.32	17.16	2.78e-15	-1.0e-17	-2.86e-18	1	-2.19e-18	1

Table 4.1: Camera parameters for Fig 4.8, Fig 4.9 and Fig 4.10

Because of the limited time, we could not perform experiments for user evaluation of TICC system, but we believe that the multi-finger gestures we proposed to interactively control a camera in 3D environment are natural to use and easy to learn. Experiments need to be further conducted.

Chapter 5

Perspectives and Conclusion

5.1 Perspectives

The basic results of our Frame metaphor and TICC system can be extended for other camera models and applications. In this section we briefly outline some possible extensions and applications of frame metaphor and TICC system.

- Collision and occlusion free navigation is an important and desirable property of any camera control system in virtual environment. Hence, TICC system can be extended by adding these features in it.
- More complex cinematographic rules can be associated with TICC for filming the scene in complex and interactive environments.
- The Frame-based interaction metaphor can be explored to use it for inspection and object manipulation tasks and can be extended to use it for remote object manipulations.
- The use of large sized multi-touch and multi-user surfaces have received special attention for solving many complex problems. One of the idea is to use *frame metaphor* for the applications like *urban planning* in which multiple users can work simultaneously and in collaborative mode using multiple cameras and multiple view ports for navigation, object inspection and object manipulation task.

5.2 Conclusion

We proposed a frame-based interactive metaphor for controlling the camera in virtual environment. This metaphor allows the user to navigate freely in virtual environment with controlling all 7DOFs of a virtual camera. With this metaphor user can also select and manipulate 3D objects in virtual environment. It helps the user to establish the relationship between mental image of the scene and the view generated by the camera.

We proposed how the multiple finger gestures can be mapped into camera parameters and established the mapping between user input and cinematographic operations with reference to which new camera parameters can be computed. We built the TICC solution using *through the lens camera control* approach proposed by Kyung et al[8, 16]. We also implemented the TICC system using direct camera control approach.

We have identified a number of issues for interactive camera control on multi-touch surfaces such as mapping of gestures, adequation with tasks and cinematographic control of camera. We believe this approach to camera control with multi-touch surfaces opens exciting perspectives in terms of a more high level and intuitive control of cinematographic trajectories.

Bibliography

- [1] Nui group - natural user interface group www.nuigroup.com.
- [2] wikipedia.
- [3] www.tuio.org.
- [4] Rami Ajaj, Frédéric Vernier, and Christian Jacquemin. Follow my finger navigation. pages 228–231, 2009.
- [5] Chris. CONINX Karin DE BOECK, Joan. RAYMAEKERS. Are existing metaphors in virtual environments suitable for haptic interaction. *Proceedings of the 7th International Conference on Virtual Reality (VRIC2005)*, pages p. 261–268., 2005.
- [6] P. Dietz and D. Leigh. Diamondtouch: a multi-user touch technology. *In Proceedings of the 14th Annual ACM Symposium on User interface Software and Technology (Orlando, Florida, November 11 - 14, 2001). UIST 01. ACM, New York, NY*, pages pp. 219–226, 2010.
- [7] Pascal Guitton Fabrice Declec, Martin Hachety. Tech-note: Scruticam: Camera manipulation technique for 3d objects inspection. *IEEE Symposium on 3D User Interfaces*, pages pp. 19–22, 2009, 2009.
- [8] Michael Gleicher and Andrew Witkin. Through-the-lens camera control. *SIGGRAPH Comput. Graph.*, 26(2):331–340, 1992.
- [9] Jefferson Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. pages 115–118, 2005.
- [10] Mark Hancock, Sheelagh Carpendale, and Andy Cockburn. Shallow-depth 3d interaction: design and evaluation of one-, two- and three-touch techniques. pages 1147–1156, 2007.
- [11] Mark Hancock, Thomas ten Cate, and Sheelagh Carpendale. Sticky tools: full 6dof force-based interaction for multi-touch tables. pages 133–140, 2009.
- [12] Hawkins and Grimm. Keyframing using linear interpolation of matrices. *Amy Hawkins and Cindy Grimm. ACM SIGGRAPH 2005 Posters*, 2007.

- [13] STAM J. FITZMAURICE G. KURTENBACH G KHAN A., KOMALO B. Hovercam: interactive 3d navigation for proximal object inspection. *In SI3D 05, Proceedings of the 2005 symposium on Interactive 3D graphics and games (New York, NY, USA, 2005)*, ACM Press, pages pp. 73–80, 2005.
- [14] Russell Kruger, Sheelagh Carpendale, Stacey D. Scott, and Anthony Tang. Fluid integration of rotation and translation. pages 601–610, 2005.
- [15] Andreas Kunz and Morten Fjeld. From tabletop-systems to tabletop: Integrating technology into interactive surfaces. *To appear in C muller-Tomfelde(Ed.) Springer LNCS 6033*, pages pp. 53–72, 2009.
- [16] Kim M. Kyung, M. and S. J. Hong. A new approach to through-the-lens camera control. *Graph. Models Image Process. 58, 3 (May. 1996)*., pages pp. 262–285, 1996.
- [17] Jun Liu, David Pinelle, Samer Sallam, Sriram Subramanian, and Carl Gutwin. Tnt: improved rotation and translation on digital tables. pages 25–32, 2006.
- [18] J.-M. Normand M. Christie, P. Olivier. Camera control in computer graphics. *Computer Graphics Forum*, vol. 27, no 8:p. 2197–2218., 2008.
- [19] S. Knodel P. Guitton M. Hachet, F. Declé. Navidget for easy 3d camera positioning from 2d inputs. *IEEE Symposium on 3D User Interfaces*, pages pp.83–89, 2008.
- [20] Anthony Martinet, Géry Casiez, and Laurent Grisoni. 3d positioning techniques for multi-touch displays. pages 227–228, 2009.
- [21] Sung Yong Shin Myoung-Jun Kim, Myung-Soo Kim. A compact differential formula for the first derivative of a unit quaternion curve. *Technical report CS CG-94-005 Department of computer science, POSTECH*, 1994.
- [22] Garber M.-Lin M. C. Salomon, B. and D Manocha. Interactive navigation in complex environments using path planning. *In Proceedings of the 2003 Symposium on interactive 3D Graphics (Monterey, California, April 27 - 30, 2003).I3D '03. ACM, New York, NY*., pages pp. 41–50., 2003.
- [23] OSBORNE S. WARE C. Exploration and virtual camera control in virtual three dimensional environments. *SI3D 90, Proceedings of the 1990 symposium on Interactive 3D graphics (New York, NY, USA)*, pages pp. 5–183, 1990.
- [24] D. Xiao and R Hubbard. Navigation guided by artificial force fields. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Los Angeles, California, United States, April 18 - 23, 1998). C. Karat, A. Lund, J. Coutaz, and J. Karat, Eds. Conference on Human Factors in Computing Systems. ACM Press/Addison-Wesley Publishing Co., New York, NY, 9-186.*, 1998.

- [25] R. Zeleznik and A. Forsberg. Unicam - 2d gestural camera controls for 3d environments. *In Proceedings of the 1999 Symposium on interactive 3D Graphics, I3D '99. ACM, New York, NY.,* pages 169–173, April 26 - 29, 1999.