Through the Lens of 25 Years: Through-the-Lens Camera Control Revisited

Michael Gleicher
Department of Computer Sciences
University of Wisconsin Madison
Caveat

I was asked to talk about old stuff

It seems arrogant to say something you did was really important or inspiring

I’ll let you judge

It continues to inspire me

I’ll mention 3 unpublished papers
(2 to appear, 1 to be written)
Through-the-Lens Camera Control

Michael Gleicher and Andrew Witkin
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA
[gleicher|witkin]@cs.cmu.edu

Abstract
In this paper we introduce through-the-lens camera control, a body of techniques that permit a user to manipulate a virtual camera by controlling and constraining features in the image seen through its lens. Rather than solving for camera parameters directly, constrained optimization is used to compute their true derivatives based on desired changes in user-defined controls. This effectively permits new controls to be defined independent of the underlying parameterization. The controls can also serve as constraints, maintaining their values as others are changed. We describe the techniques in general and work through a detailed example of a specific camera model. Our implementation demonstrates a variety of useful controls and constraints and provides some examples of how these may be used in composing images and animations.

Keywords: camera control, constrained optimization, interaction techniques

1 Introduction
Camera placement and control play an important role in image composition and computer animation. Consequently, considerable effort has been devoted to the development of computer graphics camera models. Most camera formulations are built on a common underlying model for perspective projection, under which any 3-D view is fully specified by giving the center of projection, the view plane, and the clipping volume. Within this framework, camera models differ in the way the view specification is parameterized. Not all formulations are equivalent—some allow arbitrary viewing geometries, while others impose restrictions. Even so, alternative models can be viewed to a great degree as alternative slicings of the same projective pie.

How important is the choice of the camera model's parameterization? Very important, if the parameters are to serve directly as the controls for interaction and keyframe interpolation. For example, the popular LOOKAT/LOOKFROM/VUP parameterization makes it easy to hold a world-space point central in the image as the camera moves without tilting. To do the same by manually controlling generic translation/rotation parameters would be all but hopeless in practice, although possible in principle.

The difficulty with using camera parameters directly as controls is that no single parameterization can be expected to serve all needs. For example, sometimes it is more convenient to express camera orientation in terms of yaw, elevation and tilt, or in terms of a direction vector. These particular alternatives are common enough to be standardly available, but others are not. A good example involves the problem, addressed by Jim Blinn[3] of portraying a spacecraft flying by a planet. Blinn derives several special-purpose transformations that allow the image-space positions of the spacecraft and planet to be specified and solves for the camera position. The need for this kind of specialized control arises frequently, but we would rather not face the prospect of deriving and coding specialized transformations each time they do.

In short, cameras models are inflexible. To change the controls, one must either select a different pre-existing model or derive and implement a new one. If this inflexibility could be removed, the effort devoted to camera control could be reduced and the quality of the result enhanced.

In this paper, we present a body of techniques, which we call through-the-lens camera control, that offer a general solution to this problem. Instead of a fixed set, the user is given a palette of interactive image-space and world-space controls that can be applied "on the fly" in any combination. For example, the image-space position of an arbitrary world-space point can be controlled by interactive dragging, or panned while other points are moved. Image-space distances, sizes, and directions can also be...
Through-The-Lens Camera Control

Michael Gleicher
Andrew Witkin

School of Computer Science
Carnegie Mellon University
May, 1992
Some context...
August 1991, Yosemite
Driving back to Pittsburgh
A Conversation with my Advisor
The Monday after Thanksgiving, 1991

Mike: shouldn’t the inertia tensor of a camera involve what it sees?
Andy: (math involving projections and integrals over images)
Mike: can’t we just approximate that by summing over a few points?
Andy: yeah, but why would you want to?
Mike: I could use those image points as controls for the camera
    (pause)
Andy: What are you doing for the next six weeks?

I had been building Bramble for a while. A lot of infrastructure was in place. TTL really was not a 6 week project
What was I trying to do?

And finally did by the end of my thesis in 1994.
An Application:

An example 3D application illustrating how the approach can be employed.
What I thought I was doing

My thesis: A Differential Approach to Graphical Interaction

Through-the-Lens Controls as a flexible building block
A critique

Looking back after 25 years... is it still any good?
The Math...

Pose controls as non-linear constraints

Differential approach – steps towards the goal
Solve linear problem on each step
Hacky active set methods for inequalities

Derivatives of normalized Quaternions

Key good idea – and required!

Computers are faster
Optimization is better
Just solve the NLP
Use a real QP/NLP solver

Not so bad – maybe use exponential maps instead
The math hasn’t changed much

I thought real non-linear solvers were
  too unreliable
  too slow
  too hard to implement
If I had to do it again today...

Fixed step sizes vs. solve and go to solution
  “Fixed sized steps” doesn’t give dynamics
  Probably better to solve and interpolate (or just track mouse)
Use a real QP solver per step, and probably SQP to get solution
Normalized Quaternions vs. Exponential maps
  Exponential maps are “right”
  Normalized Quaternions hacky – but integrate with numerics
Implementation would be completely different
  dense solves, simpler automatic differentiation, ...
The Implementation

Automatic differentiation
Took a while to catch on, but now good Python libraries

Object-oriented math “blocks”
Lots of caching, deferred computation
Symbolic “compiler”
Computers have changed
This doesn’t make sense

Sparse data structures
Problems are too small – modern architectures like dense matrices
Computers have changed
A lot
What was I computing on in 1991?

SGI Personal Iris

20mhz MIPS R3000
MIPS R3010 FPU
16MB RAM

16 MIPS, 1.6 MFLOPS
For that video?

SGI 4D/210GTX

Basically...
Same CPU/GPU
Better graphics card

Video I/O
It's more than just 1000x Faster

- Processor vs Memory Performance

CPU-DRAM Gap

1980: no cache in microprocessor;
1995 2-level cache
# Its more than just 1000x Faster

<table>
<thead>
<tr>
<th>1991 – MIPS R3000</th>
<th>2016 – Intel Skylake (per core)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20-25 MHZ</td>
<td>3.2 – 4 GHZ (100x or more faster)</td>
</tr>
<tr>
<td>1 integer instruction per cycle</td>
<td>Multiple integer issue</td>
</tr>
<tr>
<td>FP Multiply &amp; Add – (muti-cycle)</td>
<td>Vector FP issue (32+ per cycle!)</td>
</tr>
<tr>
<td>Memory costs 2-3 cycles (hidden)</td>
<td>Memory costs huge</td>
</tr>
<tr>
<td>Cache miss about 10-20 cycles</td>
<td>Cache misses take forever</td>
</tr>
</tbody>
</table>
About that demo

The match-moving demo was important

Caught people’s attention
Interactive tools inspired people

Match-moving is a big business!
How does it work today?

“Gold Standard Algorithm”

Use the “linear method” to get initial guess

Non-linear optimization to match control points
How off was I?

“Gold Standard Algorithm”

Use the “linear method” to get initial guess

Non-linear optimization to match control points

Interaction means we have a good initial condition!

This is actually what I did!
Residuals of image points
The linear method

For the 11 entries of the camera matrix
For more than 5 points (need enough to fully determine)
There is a linear solution to the problem!
(least squares for more than 5 ½ points)

But...
Can’t constrain to “real” camera (e.g. square pixels, simple projections)
Measures error in wrong space (so small errors mean big problems)
But this was known in 1991 (just not by me!)

Faugeras’ paper was 1986
Faugeras’ book was 1993
To combine computer-generated objects into a photographic scene, it is necessary to render the objects from the same point of view as was used to make the photo. This gem describes an iterative technique for correlating view parameters to a photograph image. The method is implemented in C (Appendix 2) as a user function with a simple driver program for testing. The following sections describe the math behind the iterative technique, some specifics about the given implementation, and an example.

This method requires that at least five points are visible in the photo image, and that the 3D coordinates of those points are known. Later, when a computer-generated object is modeled, it must be in the same 3D space as the photo objects, and must use the same units. It is not necessary for the origin to be visible in the photo image, nor does it assume a particular up direction.

For each of the five (or more) data points, the 2D screen point must be found. This can be done simply by examining the photo image, or by employing more advanced image processing techniques. Because this is an iterative technique, the 2D point need not be accurate to sub-pixel detail. The final set of view parameters will project the given 3D points to 2D locations that have the minimum error from the given 2D screen points.

In addition to the data points, an iterative process needs a starting value. A set of view parameters must be given that approximate the correct answer. The method is extremely forgiving; however, it does help if the starting eye position is at least in the correct octant of 3-space, and the direction of view looks towards the center of the 3D data.
The basics of Match-Moving in 1991

About the same time as my paper

Has the same basic ideas (uses Newton iteration, works out derivatives)

Also works out the derivatives

Thanks to:
Doug Roble (Academy Award in 1998 for Match Moving)
Mohit Gupta (who teaches our Vision class, and pointed me at the HZ book)
But Match Moving wasn’t the point...

Maybe it should have been?
Specify by TTL
Specify (match-real image)
Specify (crazy controls)
Specify (control objects)

Specify by Motions
Constrain Motions
Constrain Motions
Tracking
TTL Points as Interface

Is this a good interface?

Other than match-moving...
Do real people have problems this hard?

Not clear if anyone could use this...

(algorithms can!)
TTL Controls Beyond Points...

Mix and Match!

Who thinks that way?
Can people figure out what to use?

Maybe for declarative specification
TTL Controls to Manipulate Objects

Even I couldn’t control this well

More for interaction prototyping

Sketch-based controls get this right
Avoids 3D inferences when sketching

Or, if you do, be very careful gestural, stylized, …
TTL specification of Motion

Hard to do for interesting motions
Very non-intuitive

Probably OK in special cases

Hard to get dynamics right
TTL controls during motion...

Seems useful
Might be overkill
TTL Constraints for moving objects

This seems more useful...
Not clear how generality is needed
Constrained Optimization is a great hammer
Good parameters != Good controls (so decouple!)

A single control is OK
Mixing-and-Matching controls, less OK (as a UI)
  Understand what a control does?
  Understand what combinations will do?
  Knowing how to choose from a broad palette?

Useful to talk in terms of what you see (controls) not what causes it
A little reflection... What can I learn?

1. Have a Killer Demo!
2. Solve a problem people have
3. Provide a solution that people can use
4. Provide a vision beyond 1,2,3

The cool problem you want to solve may not be The real problem people want a solution to
Two kinds of papers (there are others)

Have a novel (and good) problem – and a “good enough” solution
Have an existing problem – and have a “better” solution

Evaluation: how to convince people:

1. Problem is good
2. Solution is good enough
   or
1. Solution is better
Lesson 2 on Evaluation

• Avoid evaluation. It is difficult, takes lots of effort, and is not always interesting.
• Take a short-cut.

Recipe for rejection?

What reviewers write:
The paper is not acceptable because a controlled user-study is missing.

What reviewers mean:
I don’t believe this work makes any sense at all, but be my guest to convince me otherwise.

Develop new methods that are so awesome, cool, impressive, compelling, fascinating, and exciting that reviewers, colleagues, users are totally convinced just by looking at your work and some examples.
Good problems lead to better solutions

Motion Retargeting -> Lee&Shin ’99 Hierarchical Approach
Motion Graphs -> Reinforcement Learning Approaches
Parametric Motion Graphs -> Motion Fields
Image Retargeting -> Seam Carving, Image Deformation Approaches, ...
Video Retargeting -> [whole literature]

Re-Cinematography (camera dynamics) -> L1 Minimization
Re-Cinematography (large deformations) -> Deformation-based stabilization
Critique of 2017...

CHI 2017
Honorable Mention Award

Yes, TTL inspired...

Measurable progress towards goal (fluent conversational characters)

or

Totally new thing (bi-directional gaze) (that others may do better in the future)
Has TTL done anything for you since then?
What has happened since then?

How Through-the-Lens continue to inspire me
What Mike Does...

Human **Data** Interaction (visualization)

Human **Graphics** Interaction (media authoring)

Human **Robot** Interaction (robots!)
Video Stabilization

Moving a camera requires getting the dynamics right

(extended from ACM Multi-Media 2007 best paper)
Video Stabilization

Problem:
Shaky video in, less shaky (good?) video out

Art (and Perception)
What is good camera movement?

Perception (and Art)
How can we avoid impossible computer vision?
Three Projects

Re-Cinematography
  What can you do beyond removing jitter?

Stabilization by 3D Warping
  How can you make bigger changes?

Stabilization by Subspace Constraints
  How do you make it practical?

Work with Feng Liu and Adobe
Lessons from Filmmaking 101

Direct the viewer’s **attention**

Use a **tripod**! (a damped one)

Avoid **distracting** the viewer (unless...)

Camera movements should be **motivated**

**Smooth** movements to make connections
The key insight:
Translate cinematography to implementation

Motion should be intentional
- Static shots should be static
- Moving shots are goal directed
  - Constant velocity with ease in/out
What the art of cinematography tells us about camera motion

Camera motions should be intentional
  – Avoid movement if not necessary
  – Move in directed ways

Re-Cinematography:
Post-process video clips so that the camera motions appear to better follow the rules.
What paths do we want?

1. Preserve the intent of the source
2. Obey the rule of cinematography:

Camera motion should be intentional
Re-Cinematography “Works”
Velocity profiles meet goals

![Diagram showing velocity profiles for source and result videos.](image)
Static segments are static
Moving segments have piecewise constant velocity
Ease in and out
Two Big Problems...

The motion “planning” is quite hacky – and can’t trade off quality
Answer: L1 optimal paths (see also our paper)

The large deformations distort horribly
Answer: Liu et. al 2009, 2011
Content-Preserving Warps (2009)
Subspace Methods (2011)
More problems...

To swing or not to swing...
Stylistic choices?

Build smart systems
no “right answer”
depends on...
taste, judgment, style,…
tended message

Avoid making decisions
Another Application...
How do we use Visual Simulation as a Behavioral Research Tool?

How to review Virtual Environment Experiences?

Head Tracked Virtual Experiences

What did they see?
Where were they “looking”?
A (3D) Video Stabilization Problem?

Easier than 3D Stabilization:
Have camera path
Have world geometry (to render novel views)

Higher Expectations!
More power for analysis
More flexibility for re-synthesis
Key Ideas

Cinematography Model
from Re-Cinematography
stable points + interpolations

Content-Dependent Metric

Segmentation

Path Generation

Reproject Original Viewpoint
Key Ideas

Cinematography Model

Content-Dependent Metric
- cameras see the same things?
- efficient GPU implementation

Segmentation

Path Generation

Reproject Original Viewpoint
Key Ideas

Cinematography Model

Content-Dependent Metric

Segmentation
  find key “fixations”
  serves as compact summary

Path Generation

Reproject Original Viewpoint
Key Ideas

Cinematography Model

Content-Dependent Metric

Segmentation

Path Generation
  - cinematic smooth arcs
  - exp. Coord interpolation

Reproject Original Viewport
Key Ideas

Cinematography Model

Content-Dependent Metric

Segmentation

Path Generation

Reproject Original Viewport
Evaluation

Three Initial Studies

Objective: VR Participant Viewed Objects
Subjective: Viewer Prefers
Objective: Viewer Comprehends
A Robotics Application

Warning this is really new – unpublished, unfinished, not yet worked out, …
The Problem. . .

What to view to show when doing tele-operation

In interesting applications the view is remote
One robot watches the other...

Get a second robot
Put a camera in its hand
No need to pick “best view”
Need Through-the-Lens!
One robot watches the other...
What do you need to do this?

Through the lens controls! (keep your eye on the hand)

Good camera dynamics (don’t disorient the viewer)
Prediction (where is the user going)
Occlusion avoidance (a big problem)
Low latency control

Payoff: useful tele-operation in complicated environments
Another robotics problem
(that roboticists seem to ignore)

Rakita, Mutlu, Gleicher @ RO-MAN 2017
Start Configuration

End Configuration

Interior Poses

End-effector Trail
Summary?

You can learn some stuff from old papers
be careful, since computers change!

Old ideas inspire new ones

Thinking about how to look at things is useful
Thanks!
To you for listening.
To the organizers for inviting me.
To my mentors, students and collaborators.
To the funding sources over the years (NSF, UW, ...).

Through the lens of 25 years...
Through-the-Lens Camera Control Revisited

Michael Gleicher
University of Wisconsin Madison
gleicher@cs.wisc.edu