CS 395/495: IBMR: Image-Based Modeling and Rendering
Spring 2004 --- Syllabus

Version 1.2 Jack Tumblin; jet@cs.northwestern.edu
www.cs.northwestern.edu/~jet/

I. 2D View Interpolation:

1) How can I 'flatten out' an photo of a flat, square object?
I don't know anything about the camera, where I was, etc; all I know is that the object is flat and square, and the camera is nothing special; it's an ordinary planar-perspective camera.

2) How can I find the 'horizon line' in a picture (even if I can't see the horizon in the picture)? What is a 'vanishing point' in an image of railroad tracks, for example?

3) I know an object doesn't change size just because I photograph it. (If it did, I could see/sense the size change, even in the pictures). Yet as I walk around the object and take pictures, it always changes size in the image if measured in pixels. Is this magic, or is there something truly constant/invariant that I could measure from these pictures?(ANS: yes, the cross-ratio). What kinds of changes happen; how could I describe them mathematically?

New Tools: Cartesian vs. Projective 2D coordinates.
--Linear warps and why they look funny
(e.g. can't simulate cornfield as seen from the highway)
--P2 Defined, Ideal points,
--Points in P2, Lines in P2
--The cross-ratio
--The H matrix and its parts (H_S, H_A, H_P) defined & explored

Reading: Zisserman Handout: Chapter 1 (skip conics: 1.2.3, & marked pages)

II. Null-Space Methods

1) How can I find the H matrix from a bunch points and lines? (e.g. x,y coordinates of the corners of the square object; points of known distance along the line, 'vanishing points', horizon lines.

2) Placing points and lines manually is tedious and inaccurate! How can I fit points, lines, circles, ellipses to the planar object(s) in my picture?

New Tools:
--DLT: Direct Linear Transformation method.
--SVD: Singular Value Decomposition & its abilities
--Ellipses, circles, and other planar conic curves in P2; C^*_∞' and angle measures
--Feature-finding: corners. Read Trucco & Verri, Chapter 4.3

Reading: SVD Handout--Sonia Leach (online), Zisserman Handout on DLT (Chapter 3.1)
1.2.3 (pp. 8-11), 1.3.1 (pg. 15-16), 1.7.3(pg 32-34). Trucco: 4.3

III. Wrapping the world around a Camera

1) How can I take a 'panoramic' image with an ordinary camera?
2) How can I make a spherical mirror act like a panoramic camera?
3) How can I 'warp' from one view of a flat object to another view, as if I were flying through 3D space?

New Tools:
--'Image stitching' methods; using DLT,SVD, P2 to join images
--Mirror-ball geometry
--Image Morph (Beier,Neely 1991, View Morph (Seitz,Dyer 1996)
--Feature finding: Hough Transform, 'Snakes'--interactive, user-guidable contours...

Reading: Trucco & Verri Chapter 5. "Image Metamorphosis" Beier, Neely SIGG'92 (html-no pictures)

IV. Measuring Light with a Camera

1) What are the units of measurement for the light in one pixel?
2) How much light is "0" and "255" for a pixel in my camera?
3) Many images have both over-exposed and under-exposed areas.
How can I make an image that won't have empty white areas or black areas?

New Tools:
--Radiometry, pinhole camera/thin lens model
--Linearity of Radiance
--Radiance Maps (Debevec1997) and Tone Mapping
--Environment Mattes: incoming light direction at each pixel…

Reading: Trucco & Verri Chapter 2.2, 2.3
"Recovering High Dynamic Range Radiance Maps from Photographs" Debevec,Malik SIGG97

V. Measuring Shape with one Camera (and measuring the camera as well!).

1) How can we convert pixel distances to inches in 3D?
2) How can we convert angles measured in a photograph to 3D angles?
3) Given a photograph of a known 3D object (e.g. a set of points) how can we find WHERE the camera is?
4) Given a 3D point, how can I find the pixel that sees it?

New Tools;
--Intro to P3; points, planes, and line weirdness
--Conics for angle measurement
--Camera Calibration matrix; extrinsic and intrinsic params
--'Photometric Stereo'
--Silhouette finding; Visual Hulls.

VI. Measuring Shape with Two Cameras

1) If we two cameras see the same object, how can we find the object's shape?

New Tools:
--Fundamental Matrix
--Epipolar Planes: the search for Correspondence

VII. Measuring Everything with Infinite Cameras

1) How can we describe ALL POSSIBLE rays through a plane? Through a volume?
2) How could I measure reflectance of a diffuse object with a camera?
3) How could I measure texture? BRDF? Subsurface scattering

New Tools:
--Light Fields--two plane parameterization
--Camera/Projector duality
--Polarization for Specular highlights

Reading:

VIII Measuring Light by its Changes

1) How can we transfer a tattoo from a photo of one person to a photo of another person, without noticeable seams or color mismatches?
2) How can we make a short video sequence of a waterfall into an endless sequence that doesn't ever seem to repeat itself? (Extension of texture synthesis)

New Tools:
--Poisson Solvers: Gradient Domain Image Editing
--Graph Cuts for Video Textures

Reading: Poisson Solver Handout,….

MIDTERM (20%):

Written, take-home test to be sure you really did learn something about projective geometry and the SVD. **Assigned: April 29, 2004; Due May 6, 2004**

PROJECT(60%):

Do something you would find interesting that transforms an image or a set of images into a different image or set of images; from another view point, with different lighting, find or approximate shape, reflectance, lighting, etc.
You may use either real images, synthetic images (we'll help you render), or both.
Consider using synthetic images with known camera geometry as 'test' data to verify that your transformations of real images are correct.

<1 page Project Description due:	**Thurs April 22, 2004.**
<1 page Progress Report due:	**Tues May 11, 2004.**
In-class demo + Source Code + Demo Website due	**Tues May 25, 2004.**

Class Participation(20%)

Speak up! Ask Questions! I want people to think out loud and try to solve problems in-class. I want you to learn methods you have actually applied in this class—many people know the general idea of IBMR, but too few can write real code for it that actually works.