# Multiprojector seamless tiled display

Pranav Kant Gaur

Graphics and Visualization section

November 5, 2014

# Agenda

1. Problem statement
2. Motivation
3. System setup
4. Overview of approach
5. Algorithm
6. Novelity
7. System configuration
8. Results
9. Further work

# Problem statement

Given an arbitrarily arranged grid of projectors projecting at arbitrary positions onto the screen, find the region in each projector image buffer where if the original image is mapped onto will result in a seamless rectangular projection region on the screen.
In short, ADD UNALIGNED=¿ ALIGNED ??????????????????????

# Motivation

1. Why *Tiled*?
   An image with spatial resolution higher than that *perceivable* by human eye cannot be visualized without reducing its resolution(So actually *high megapixels* is just a marketing strategy, always check *pixels per inch* instead). We can achieve this by spatially *streching* the content.
2. Why *Multiprojector*?
   Seams of monitors used in our earlier Tiled display system were *distracting*. Projectors do not pose such limitation.

# System setup

Developed system has:

1. 3X3 grid of projectors
2. Rear projection screen
3. 1 digital camera
4. Workstations arranged in Master-slave configuration

ADD IMAGE HERE?????????????????

# Overview of approach

1. Geometric alignment
    1. Compute the share of each projector in the global image
    2. Map texture of that region to projector based on a mapping which ensures geometric continuity
2. Edge blending
    1. Determine overlapping region between neighbouring projectors
    2. Attenuate intensity of pixels in that region for all overlapping projectors so that their overlap does not create *bright* junction between them

# Algorithm

## Compute screen to camera relation

# Algorithm(contd.)

## Project and detect features for each projector

Features are mapped to screen for subsequent computation.

# Algorithm(contd.)

## Compute *local* bounding boxes

# Algorithm(contd.)

## Compute biggest *local* bounding box

# Algorithm(contd.)

## Compute *global* bounding box

# Algorithm(contd.)

1. Compute normalized coordinate for detected features
2. Compute mapping between normalized *projected* feature points and normalized *detected* feature points
3. Compute position of local bounding boxes in the global bounding box
4. Compute alpha map

# Novelity

Cross-ratio based full projection region recovery ADD FIGURE WITH CROSS RATIO WITOUT CROSS RATIO??????????? EQUATION OF CROSS RATIO???????????????????????????

# System configuration

1. Software
    1. Written in C
    2. Dependent on OpenCV(v2.4.1) and libgphoto2(v2.5.2)
    3. Works on Ubuntu(12.04 LTS) and Scientific Linux(6.1)
2. Hardware
    1. 3X3 grid of NEC 200X DLP projectors
    2. 2.4mX1.8m acyilic glass based rear projection screen(from ScreenTech,Germany)
    3. Canon Powershot G7 digital camera
    4. 4 Workstations(1 master+ 3 slave) with ???
       Each slave controls rendering on a row of projectors. It recieves rendering information from master using *Chromium* framework.

# Results

1. Alignment procedure completes in 3-4 minutes as opposed to 30 mins. consumed in manual alignment approach
2. Proposed *Cross ratio* based approach resulted in recovery of full projection region
3. Maximal misregistration between neighbouring projectors was around 2.5mm SHOW FIGURE?????????????????
4. Paper selected for presentation at IEEE ICCCI-2014.

# Further work

View independent color seamless is still an *open* problem EMBEDD A VIDEO WITH VOICE SHOWING THE PROBLEM IN THE DISPLAY Color seamlessness problem