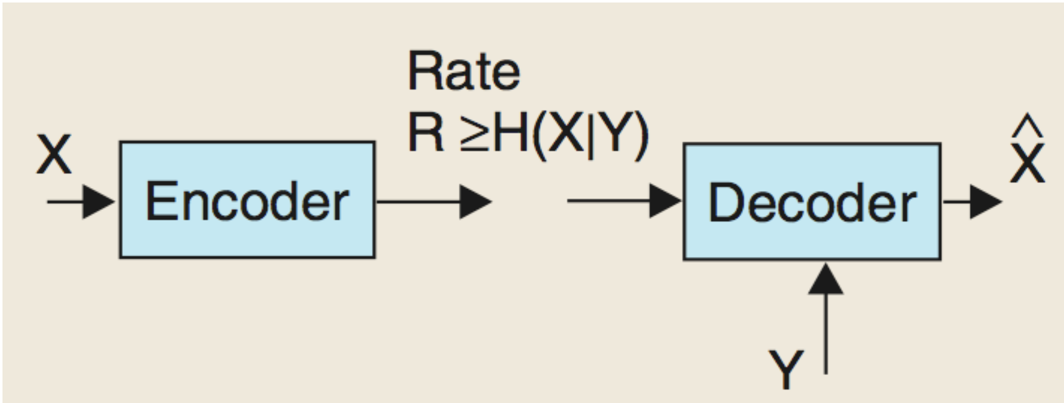
WIRELESS VISUAL SENSOR NETWORK

9/2008 – 5/2009
RESEARCH OVERVIEW

Kishore Padmaraju & Matt Magill

Motivation

- Application of distributed source coding in wireless visual sensor network (VSN)
  - Utilize inherent image redundancy by implementing side-channel encoding techniques



Wireless Mote Requirements

- Large onboard program memory and sufficient processing capabilities
  - Handle side-channel encoding
- Acceptable visual sensor connectivity and interface
- Ease of implementation
  - Pre-configured network
- Mote synchronization
  - Enable proper timing
- Acceptable data rate

Camera Requirements

- CMOS visual sensor technology
- Auto focus if possible
  - Increased picture clarity
- Auto synchronization of less than < 125μs
  - Multiple images of target
- Onboard buffer
  - In order to avoid lost information
- Communication standard to interface with mote
  - Ease of connection

VSN Possibilities

- Motes:
  - Crossbow
    - iMote2
    - mica2
  - Sentilla
  - Stanford University
    - Stephan Hengstler design
- Cameras:
  - Point Grey Research Inc.
    - Chameleon
      - (USB2.0)
    - FireflyMV
      - (1394a/USB2.0)
    - Dragonfly2
      - (1394a)

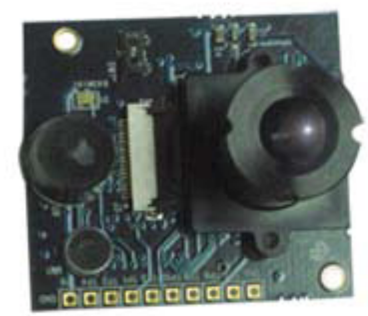
Crossbow Imote2

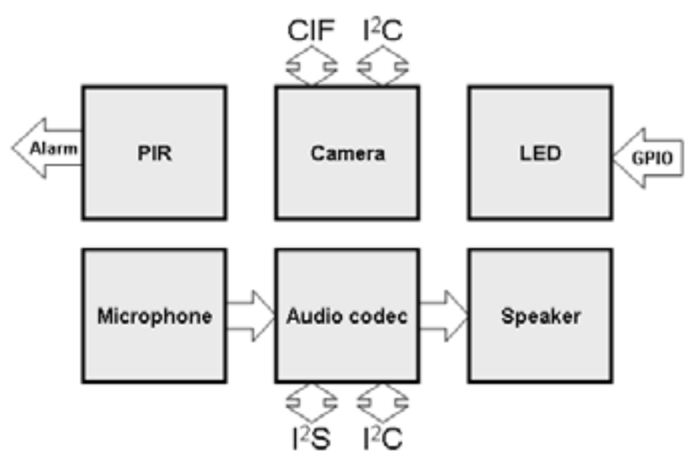
- Processor: Intel PXA271 XScale® at 13 – 416MHz
- Memory: 256kB SRAM, 32MB SDRAM, and32MB of FLASH memory
- I/O: Camera Chip Interface
- Radio: CC2420 IEEE 802.15.4 radio transceiver from Texas Instruments which supports a 250kb/s data rate with 16 channels in the 2.4GHz band.
- Antenna: Surface mount antenna with a range of ≈ 30m

Mode	Current	Remarks
Deep Sleep	390 µA	
Active I	31 mA	13MHz, radio off
Active II	44 mA	13MHz, radio Tx/Rx
Active III	66 mA	104MHz, radio Tx/Rx

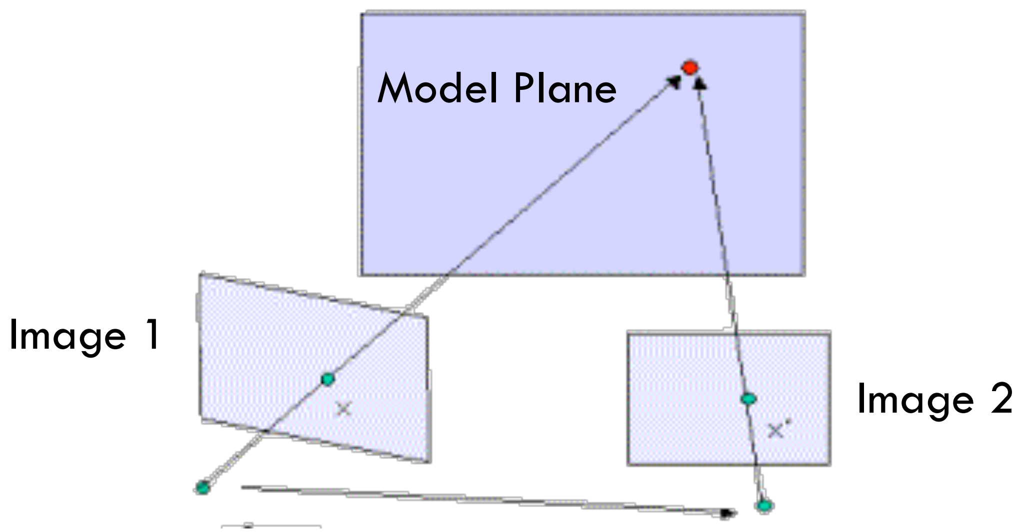
IMB400 Multimedia Board

- Color Image and Video Camera Chip
  - OmniVision OV7670
- Fair image resolution
  - Up to 640x480 pixels
  - 30 fps max
- Hardware image scaling and filtering
  - automatic exposure, gain, white balance, black level
- Image controls
  - saturation, hue, gamma, sharpness
- Easily connects to IMote2 through advanced connector set





Homography

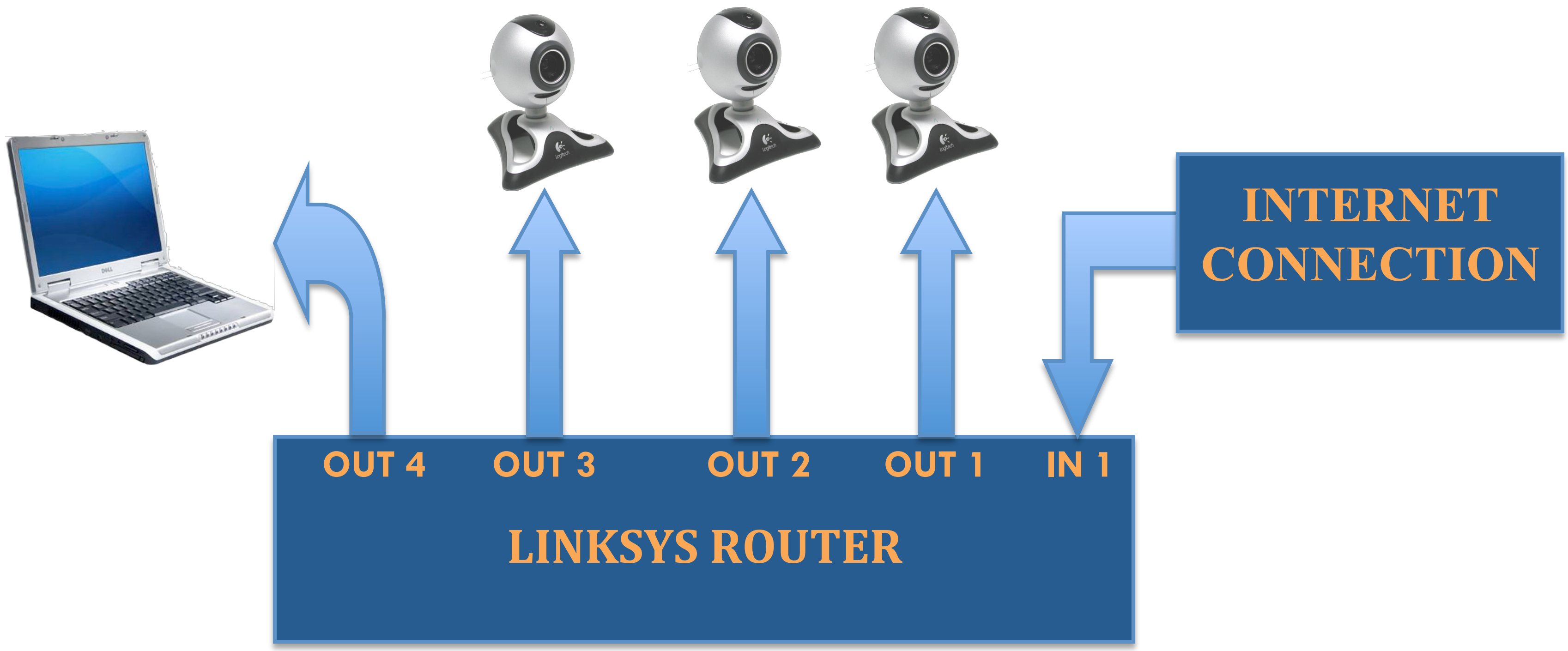


$$\tilde{m} = H\tilde{M}$$

$$H = A\begin{bmatrix} r_1 & r_2 & t \end{bmatrix}$$

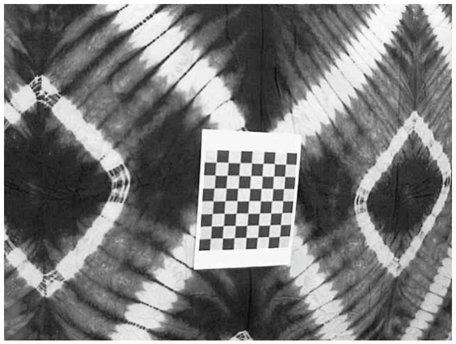
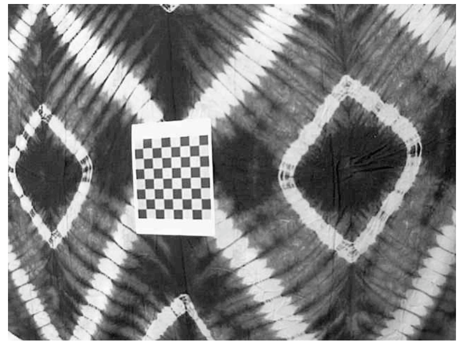
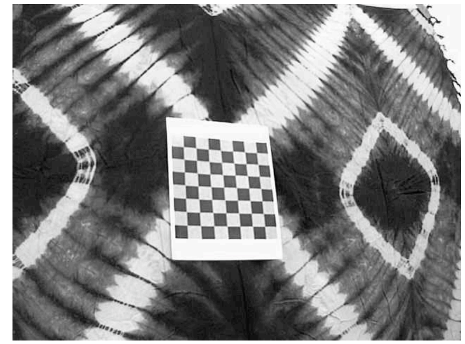
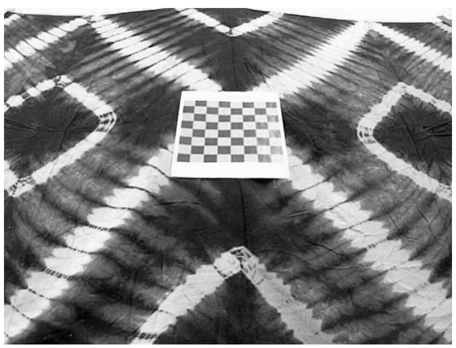
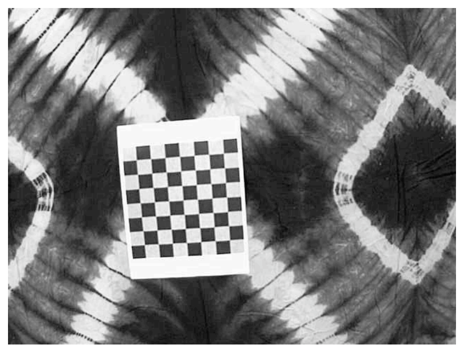
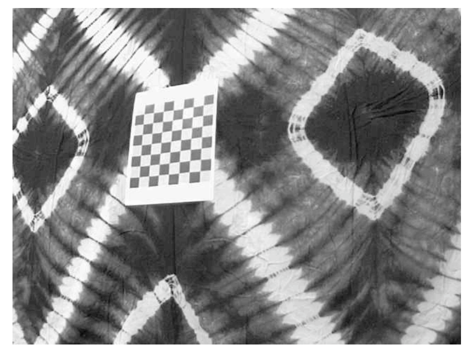
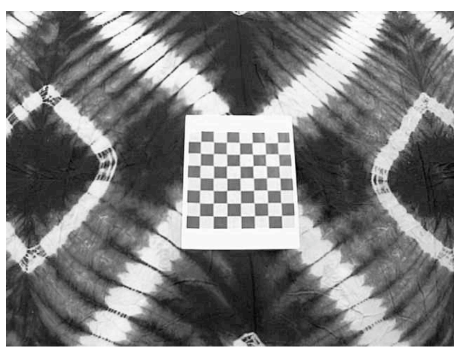
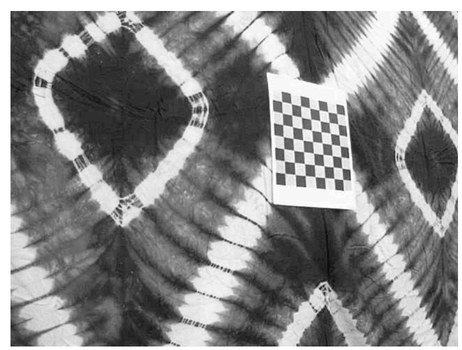
$$\mathbf{A} = \begin{bmatrix} \alpha & c & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

Image Capturing

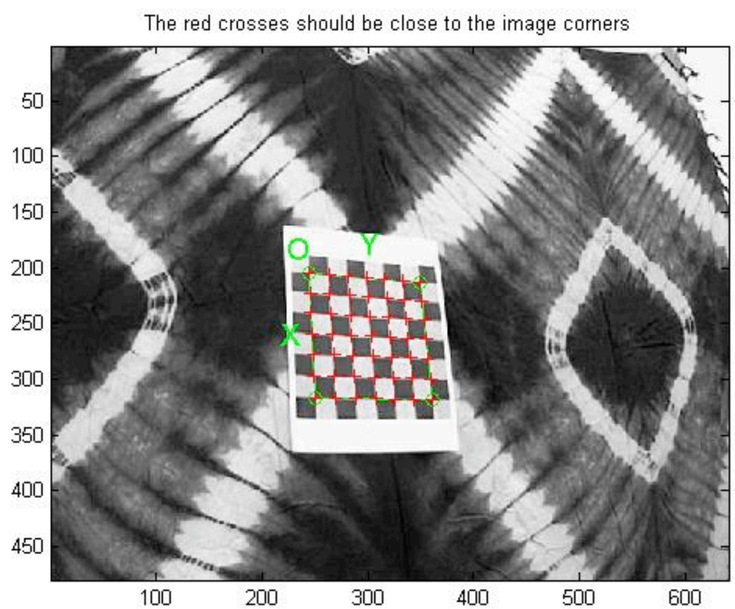


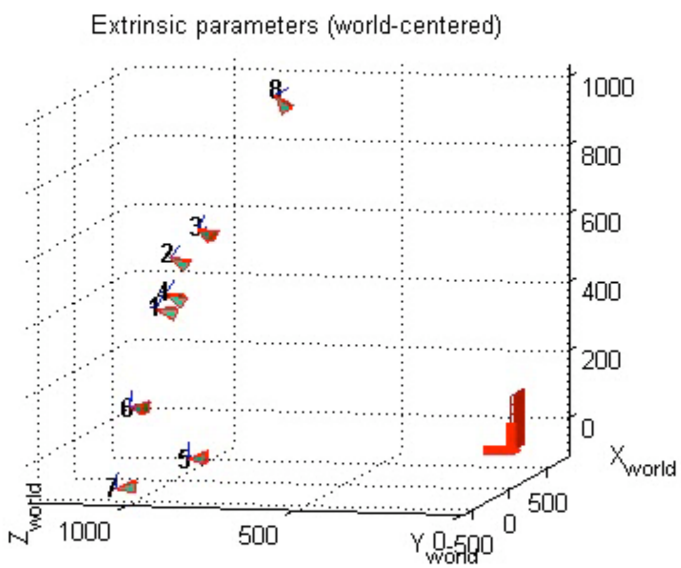
- Stanislava's Camera Manager program

Trial Images

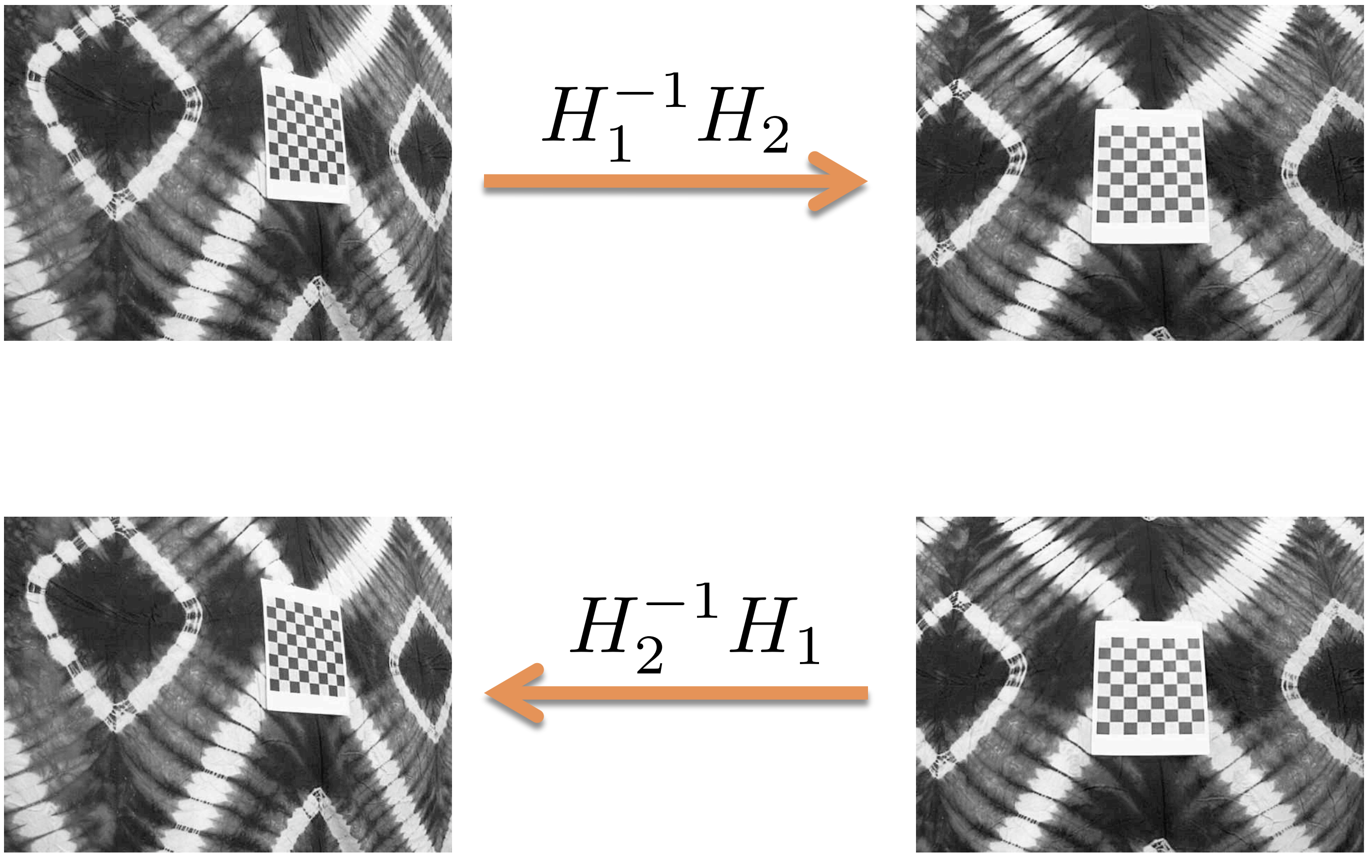
Camera Calibration Toolbox



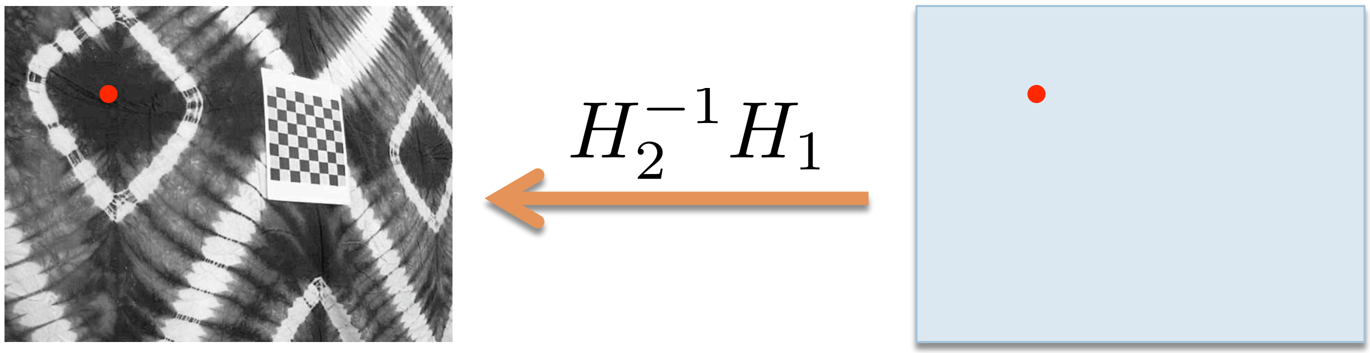


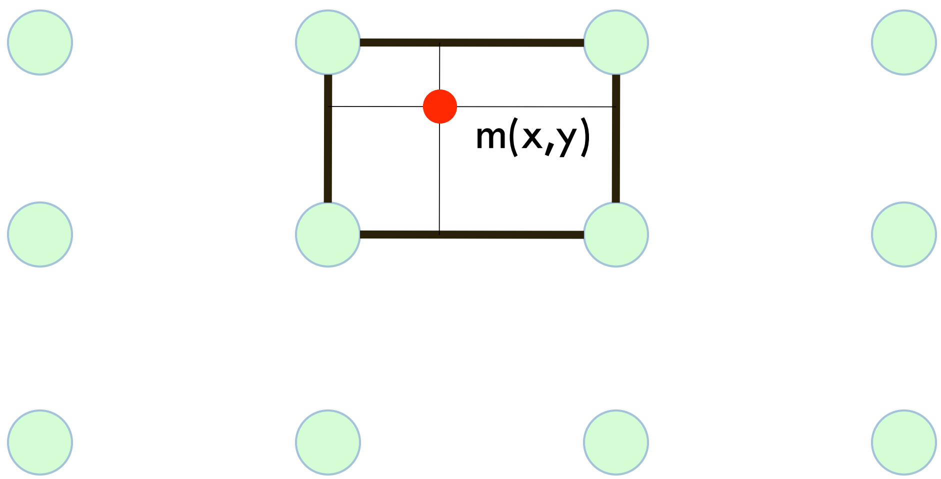
- Computes Homographies and Camera Parameters

Homography



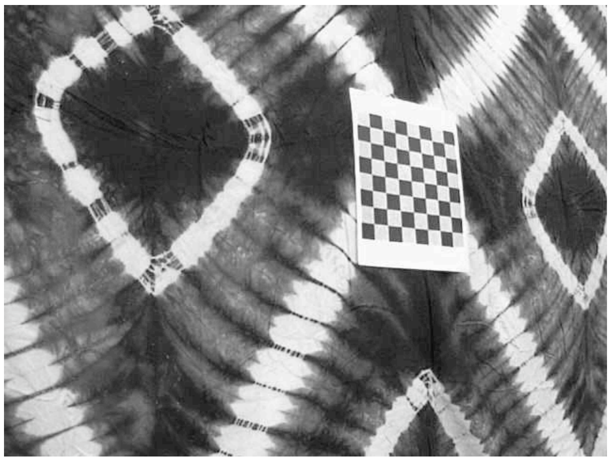
Interpolation





Initial Results

Image 1



Homography

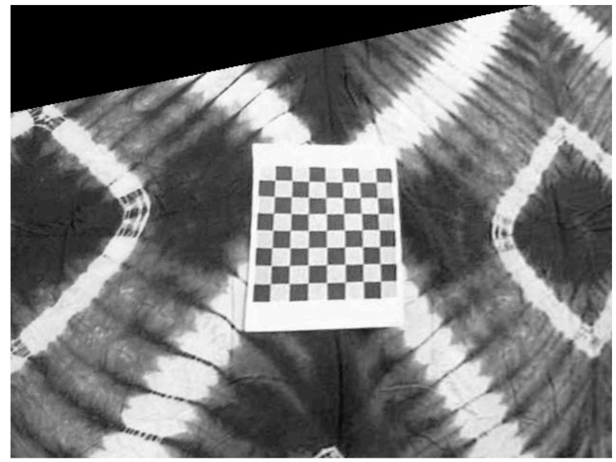
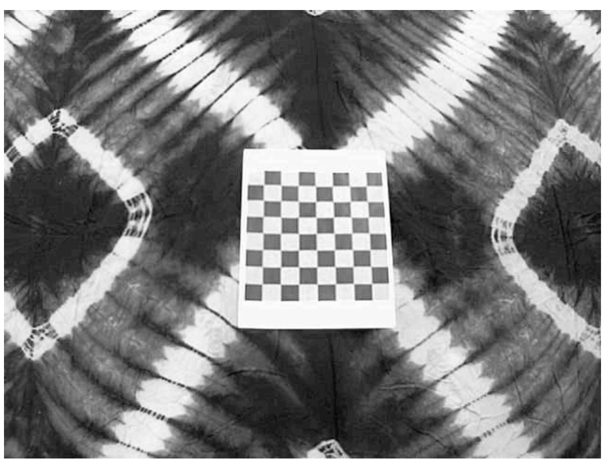
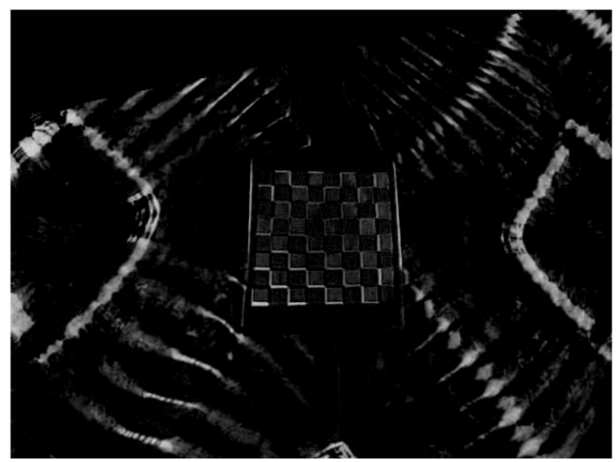


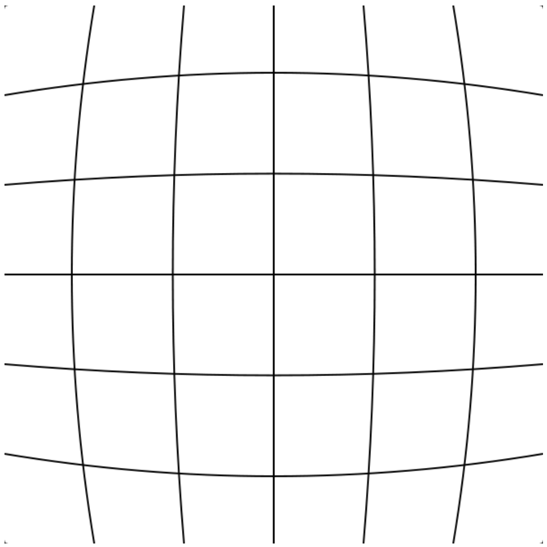
Image 2



Error



Radial Distortion

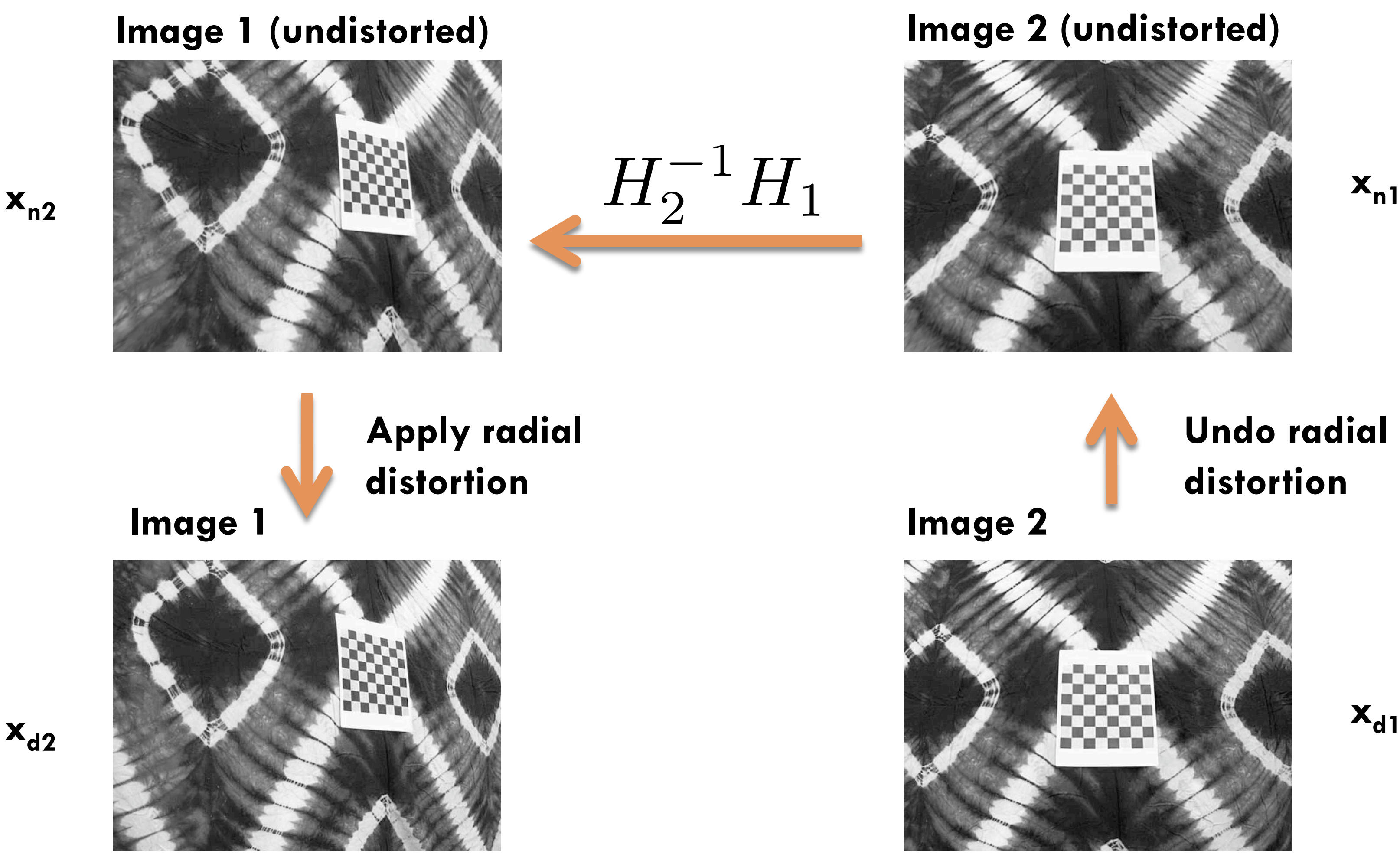


- Camera Calibration Toolbox calculates radial distortion model

$$x_d = (1 + kc(1)r^2 + kc(2)r^4 + kc(5)r^6)x_n + dx$$

x_d = distorted coordinate
x_n = normalized coordinate

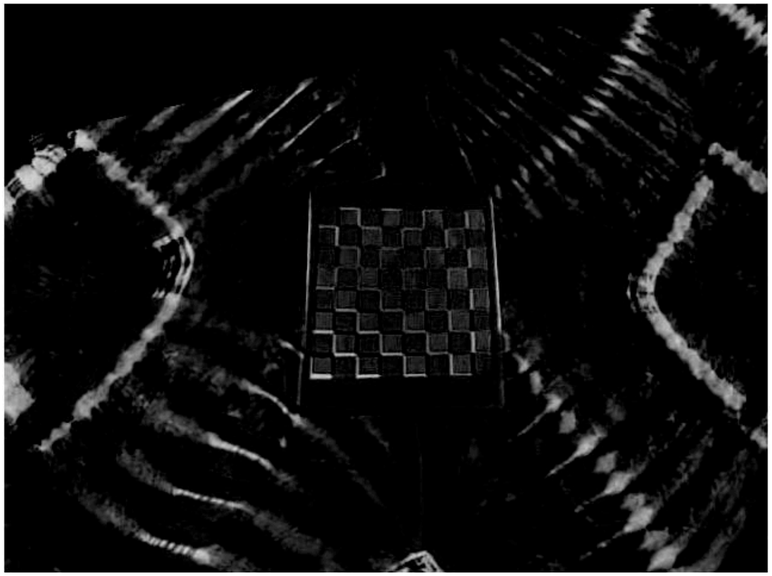
Homography with Radial Distortion



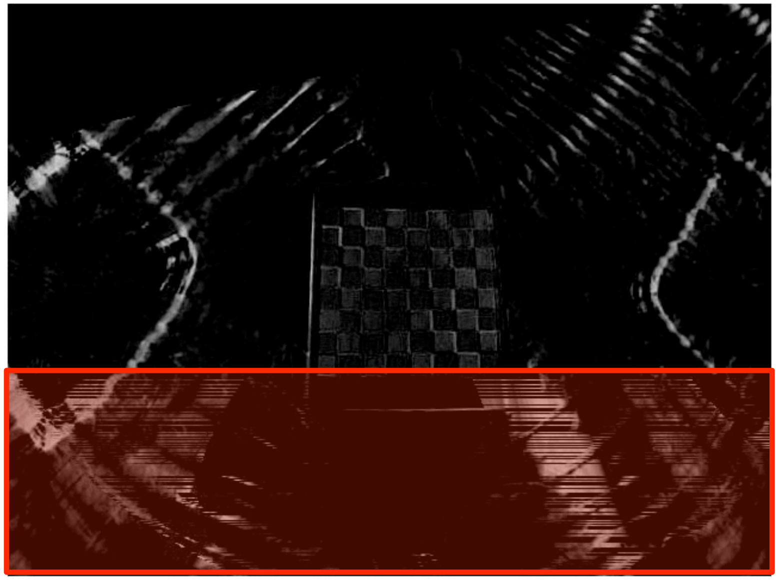
Results with Radial Distortion

- Solving non-linear equations with Matlab's fsolve()

Without radial distortion



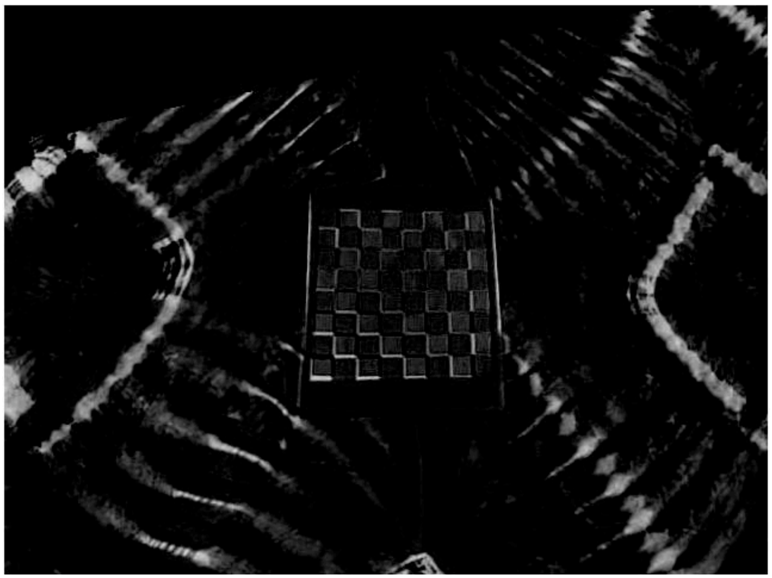
With radial distortion



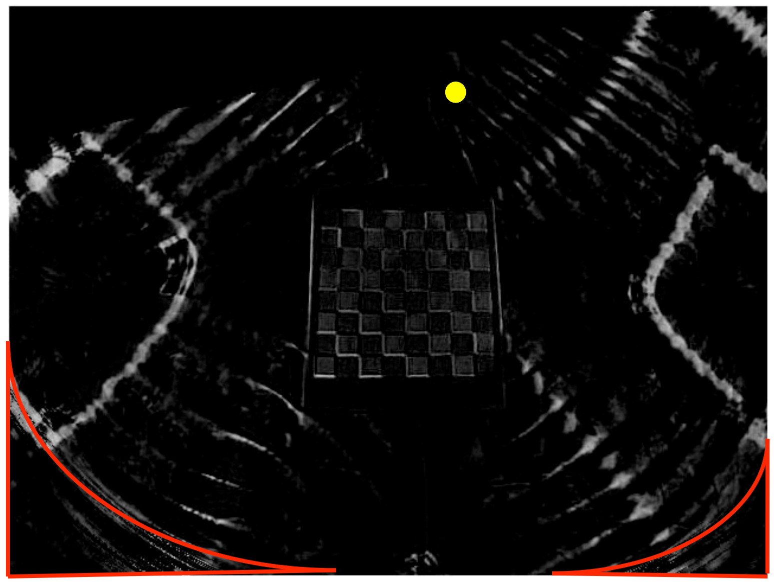
Results with Radial Distortion

- Using Camera Calibration Toolbox

Without radial distortion



With radial distortion



Future Plans

- Finish Radial Distortion Work
- Slepian-Wolfs encoding
- Gather new set of images
  - Repeat Homography work

Question?