3D Assistant for Remote Learning

CS231A Project Presentation

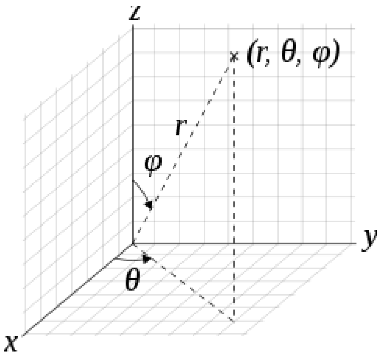
Vikas Paliwal
vpaliwal@stanford.edu
March 15, 2021

Contents

- Introduction
  - Problem, motivation, background
- Proposed Approach
  - Edge/Line/Quad Detection
  - Single View Metrology
  - Homography, Perspective Geometry –
    - Novel ULDH (Upper-Lower Decomposition of Homography) approach
- Results
  - Ground truth image dataset creation
  - Benchmarks, Efficacy of approach

Introduction

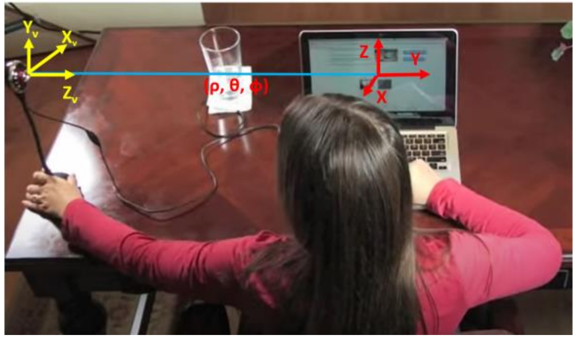
PROBLEM
MOTIVATION
BACKGROUND



Problem with Remote Learning and Exams

- Maintaining integrity of online exams/tests is challenging
- Greater need to monitor examinees' device screen and work area
- Second side camera highly recommended and used, but…

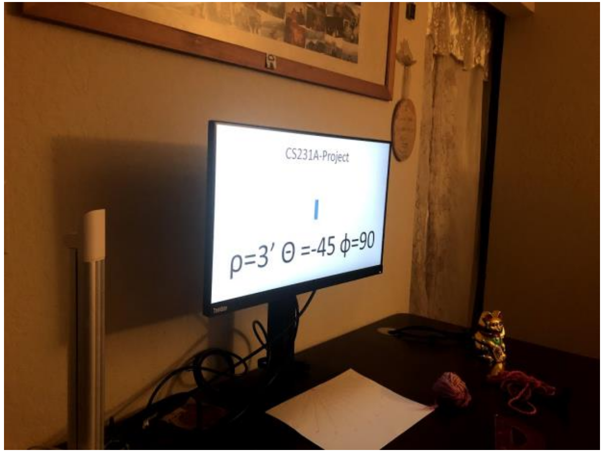
- Side view is as good as side camera placement !!!
- Need to auto-detect poor camera placement and flag automatically
- CS231A principles to the rescue ☺

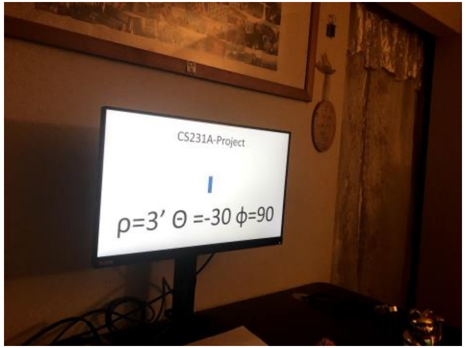


Source: Online proctoring by Kryterion

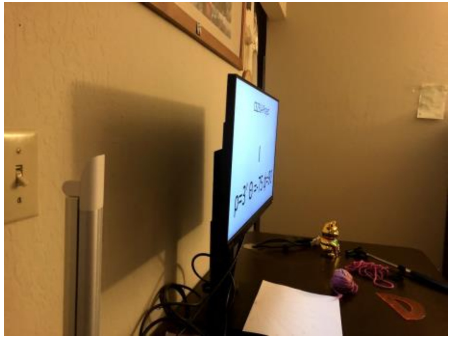
Precise Problem Statement

Given side image or video stream,





Good



Bad

Can we classify camera settings?

Using decision boundaries like,

$$20^o < |\theta| < 70^o$$

$$55^o < \varphi < 135^o$$

But how to estimate θ, φ?

Personal Motivation

Volunteer as NSB Coach – need to monitor middle/high school kids' screens from side camera; badly need continuous cam placement check

Source: NSB rules for Coaches

Background, prior work

Standard techniques exist to decipher geometry from:

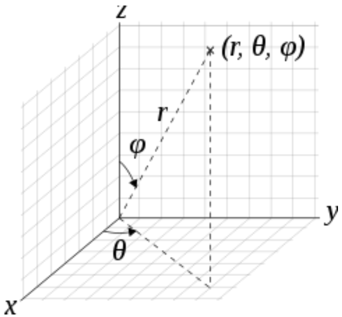
- OpenCV's *findHomography()* and *decomposeHomographyMat()* but need camera intrinsic matrix, K, for decomposition
- Other methods assume simple camera matrix form, diag(f,f,1)
- Tested both methods and ended up with noisy, unusable results

Estimating precise scene geometry from single view w/o camera intrinsic is hard but few problem-specific avenues to exploit:

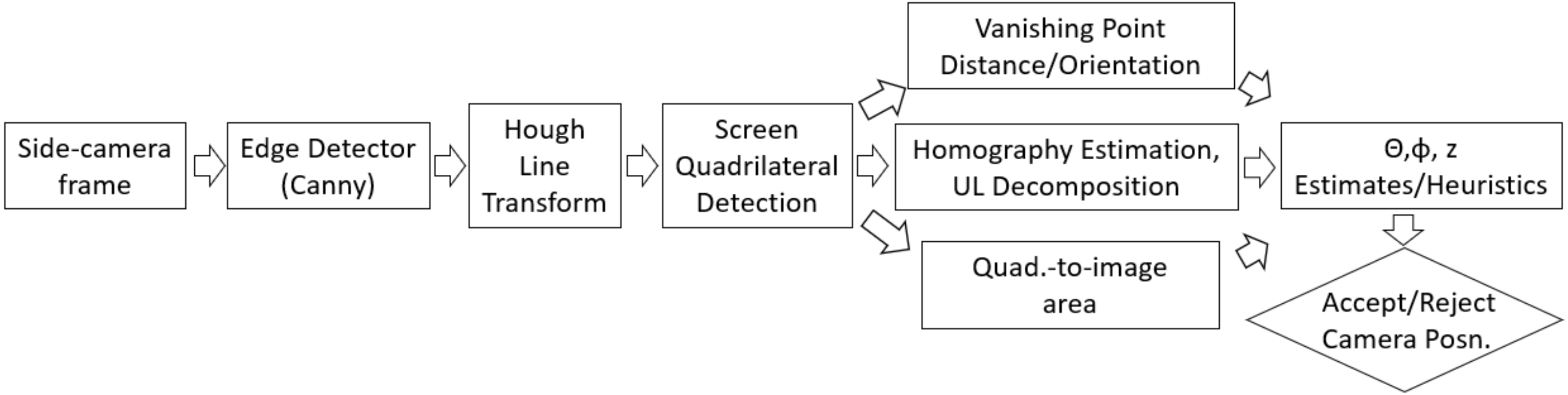
- Device screens are rectangular
- Most device screens have standard aspect ratios, 16:9, 16:10 etc.

Approach

EDGES, QUADS
VANISHING POINTS
ROTATIONS, HOMOGRAPHY



Overall Processing Pipeline



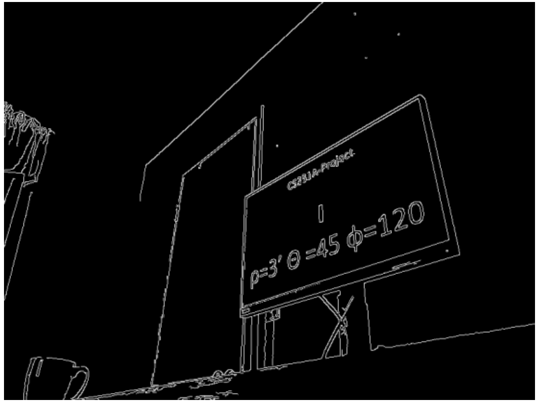
Edge, Line Detection

Edge Detection:

- Use OpenCV's Canny Edge Detector, *Canny()*

Line Detections

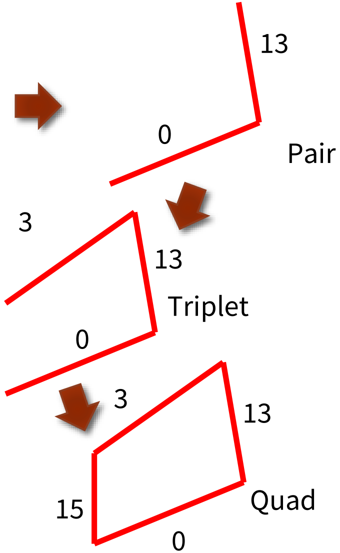
- Leverage OpenCV's probabilistic Hough Line Transforms, *HoughLinesP()*.

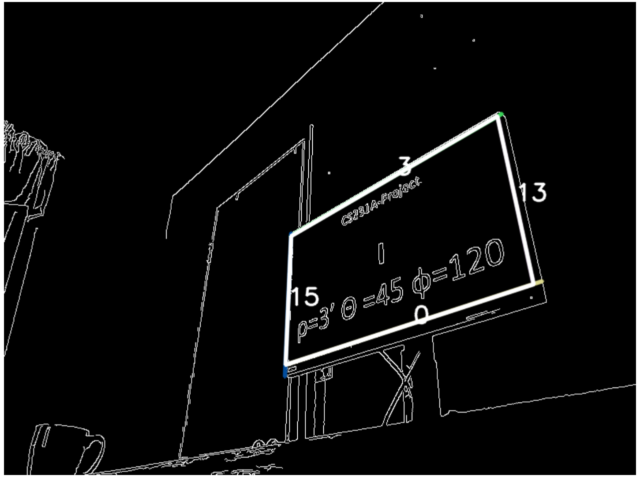


Quadrilateral Detection

From Hough lines, successively build line pairs, triplets to quads and pick a quad with:

- Largest area
- Non-enclosing (actual display area, not monitor boundaries)

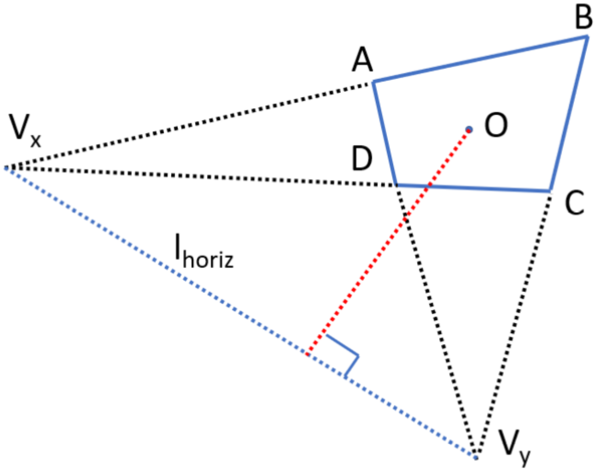


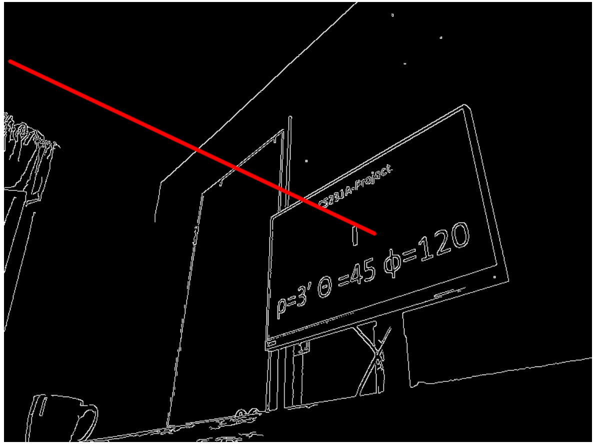


Vanishing Point Constraints

Rectangle shape offers two vanishing points (VPs) and

- Identify VP along length V_x and along width V_y from relative X/Y distance from quad center
- Line joining V_x and V_y is line at infinity and normal to it from quad center indicates plane normal in image





Vanishing Point Constraints

The locations of V_x and V_y w.r.t. quad center puts constraints on θ, φ

- To avoid border ambiguities, use the VP closer to quad center (to be used later)

V_x position

$0^o <= \theta < 90^o$

$-90^o < \theta <= 0^o$

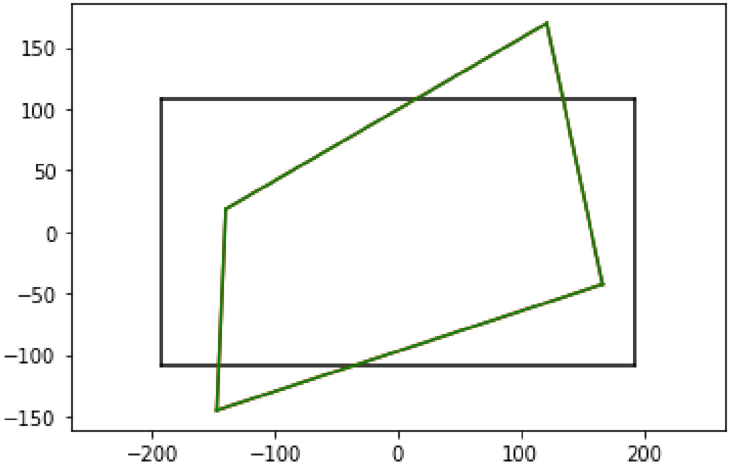
V_y position

$90^o < \phi < 180^o$

$0^o < \phi <= 90^o$

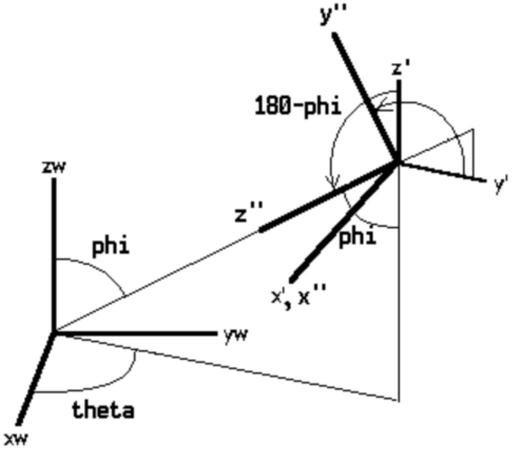
ULDH Method – Compute Homography wrt. Normal View

Using known/guessed aspect ratio, overlay a rectangle on quadrilateral and compute corresponding planar homography using OpenCV's *findHomography()*



ULDH Method – 3D Viewing Transformation (Polar Coods.)

It is understood that 3D viewing transformation (extrinsic camera matrix) of world coordinate system to camera system in terms of polar coordinates is given by,



$$P_V = \begin{bmatrix} -\sin(\theta) & \cos(\theta) & 0 & 0 \\ -\cos(\phi)\cos(\theta) & -\cos(\phi)\sin(\theta) & \sin(\phi) & 0 \\ -\sin(\phi)\cos(\theta) & -\sin(\phi)\sin(\theta) & -\cos(\phi) & \rho \end{bmatrix}$$

Source: Prof. Eckert's page

ULDH – Planar Homography in Polar Coordinates

Mapping of a device screen plane point *P* (recall X =0) to corresponding image point *p* is given by (assume simpler but realistic camera intrinsic matrix, K, square pixels, with no skew but non-zero c_x, c_y)

$$p = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = KP_vP = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -sin\theta & cos\theta & 0 & 0 \\ -cos\phi \cdot cos\theta & -cos\phi \cdot sin\theta & sin\phi & 0 \\ -sin\phi \cdot cos\theta & -sin\phi \cdot sin\theta & -cos\phi & \rho \end{bmatrix} \begin{bmatrix} 0 \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta & 0 & 0 \\ -cos\phi \cdot sin\theta & sin\phi & 0 \\ -sin\phi \cdot sin\theta & -cos\phi & \rho \end{bmatrix} \begin{bmatrix} Y \\ Z \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & c_x \\ 0 & 1 & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta & 0 & 0 \\ -cos\phi \cdot sin\theta & sin\phi & 0 \\ -sin\phi \cdot sin\theta & -cos\phi & \rho \end{bmatrix} \begin{bmatrix} Y \\ Z \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & c_x \\ 0 & 1 & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{f}{\rho} cos\theta & 0 & 0 \\ -\frac{f}{\rho} cos\phi \cdot sin\theta & \frac{f}{\rho} sin\phi & 0 \\ -\frac{1}{\rho} sin\phi \cdot sin\theta & -\frac{1}{\rho} cos\phi & 1 \end{bmatrix} \begin{bmatrix} Y \\ Z \\ 1 \end{bmatrix} = H_\pi P_{screen\_plane}$$

ULDH – Key Observation, H = UL

- Note that homography matrix in this case is a product of an upper triangular and a lower triangular matrix – we use this to decompose homography from first step into U and L matrices.

$$H_\pi = \begin{bmatrix} 1 & 0 & c_x \\ 0 & 1 & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{f}{\rho} cos\theta & 0 & 0 \\ -\frac{f}{\rho} cos\phi \cdot sin\theta & \frac{f}{\rho} sin\phi & 0 \\ -\frac{1}{\rho} sin\phi \cdot sin\theta & -\frac{1}{\rho} cos\phi & 1 \end{bmatrix} = U \cdot L$$

- Notably *numpy* offers a LU decomposition but not a UL decomposition, so an implementation is made for UL decomposition for a 3x3 matrix using Gaussian elimination/row-reduction

ULDH – Implementing UL decomposition

- For the homography H from OpenCV's *findHomography()*, we can find two matrices T_1 and T_2 using row reduction and finally get desired UL decomposition.
- Note that UL decomposition is not unique and any diagonal matrix can be inserted in between viz. $H=UL= UD^{-1}DL$ → the square pixel assumption $f_x = f_y = f$, must be true

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}, T_1 = \begin{bmatrix} 1 & 0 & -h_{13} \\ 0 & 1 & -h_{23} \\ 0 & 0 & 1 \end{bmatrix}$$

$$\implies H' = T_1 \cdot H = \begin{bmatrix} h'_{11} & h'_{12} & 0 \\ h'_{21} & h'_{22} & 0 \\ h'_{31} & h'_{32} & 1 \end{bmatrix}$$

$$T_2 = \begin{bmatrix} 1 & -\frac{h'_{12}}{h'_{22}} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \implies H'' = T_2 \cdot H' = \begin{bmatrix} h''_{11} & 0 & 0 \\ h''_{21} & h''_{22} & 0 \\ h''_{31} & h''_{32} & 1 \end{bmatrix}$$

$$U = T_1^{-1} \cdot T_2^{-1}, L = H'' \quad (2)$$

ULDH – Extracting angle relations from L

Even though focal length and camera-screen distance are unknown so (f/ρ) is not known but we can use ratios from L.

$$L = \begin{bmatrix} \frac{f}{\rho}cos\theta & 0 & 0 \\ -\frac{f}{\rho}cos\phi \cdot sin\theta & \frac{f}{\rho}sin\phi & 0 \\ -\frac{1}{\rho}sin\phi \cdot sin\theta & -\frac{1}{\rho}cos\phi & 1 \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix}$$

$$\Longrightarrow$$

$$l_{22} \cdot cos\theta - l_{11} \cdot sin\phi = 0$$

$$l_{22} \cdot cos\phi \cdot sin\theta + l_{21} \cdot sin\phi = 0$$

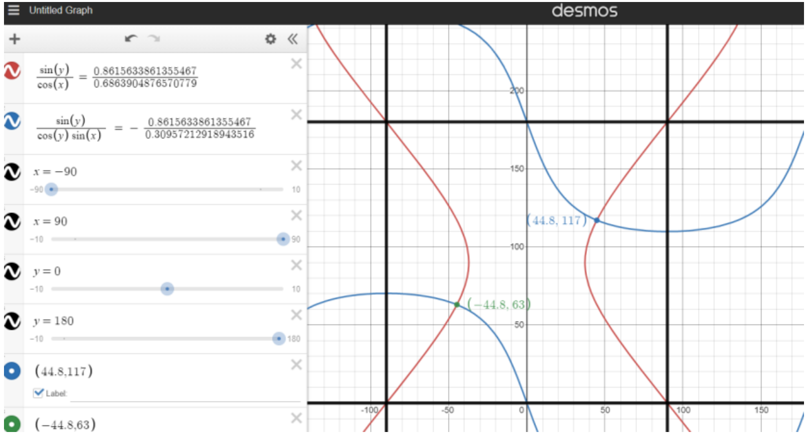
Notably, first two rows are used because more noisy results were observed from third row relations (possibly due to scaling by f)

ULDH – Solving for θ, φ

- Two equations from L are nonlinear with two variables, θ, φ and are solved using *fsolve* function available in *numpy.*
- Note that in our valid range {-90 < θ < 90,0 < φ <180}, there are two valid solutions i.e. if (θ, φ) is a solution, (- θ, 180-φ) is also a solution.
- Solution: Use vanishing points constraints to resolve ambiguity and get a unique (θ, φ) pair

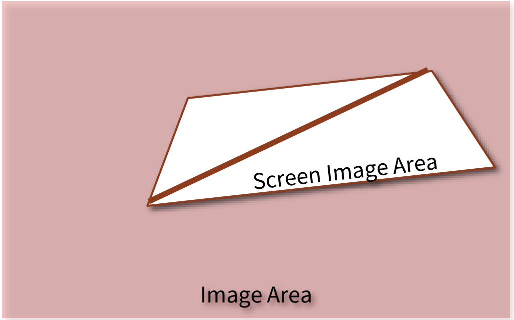
$$l_{22} \cdot cos\theta - l_{11} \cdot sin\phi = 0$$

$$l_{22}cos\phi \cdot sin\theta + l_{21} \cdot sin\phi = 0$$



Distance Heuristics

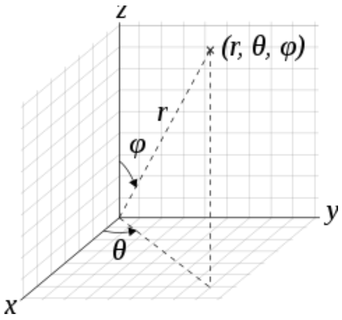
- The area of quadrilateral is computed as sum of two constituent triangles using *np.linalg.det()*
- If ratio of screen image quadrilateral area to image area (QA) is below a certain threshold, say 7.5%, reject the camera position as too far



$$QA = \frac{Screen\ Image\ Area}{Image\ Area}$$

Results

MEASUREMENT
TESTING
CONCLUSION



Measurement Tools/Procedure

Protractor, rulers

+

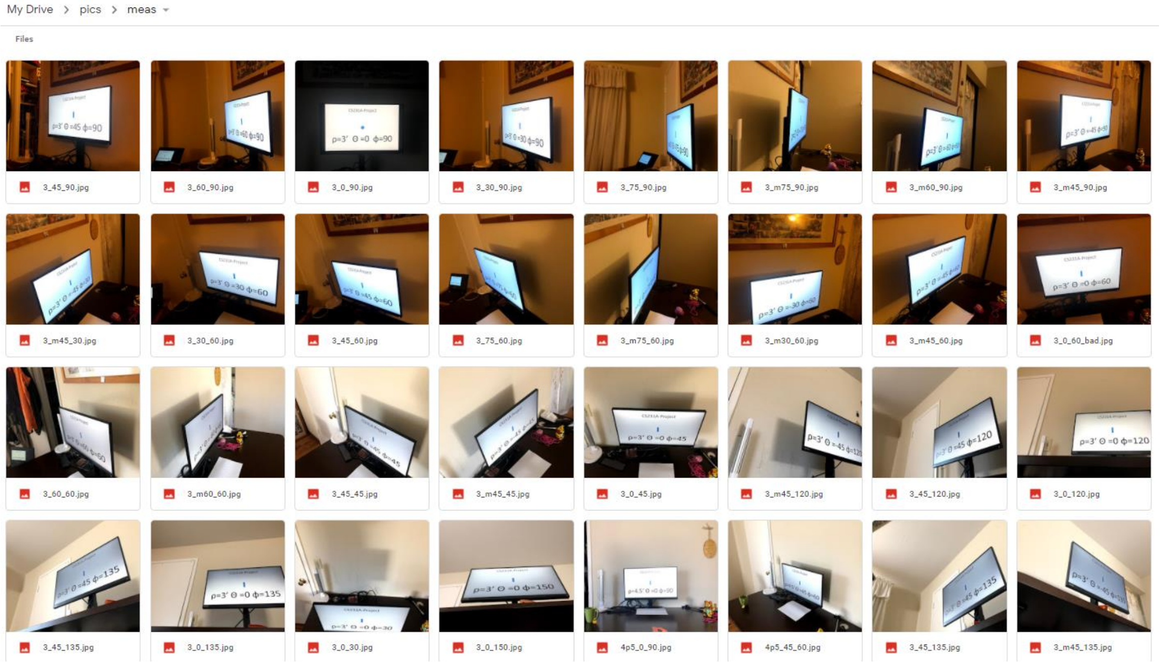
Angle Gauge

+

iPhone

Ground Truth Image Dataset

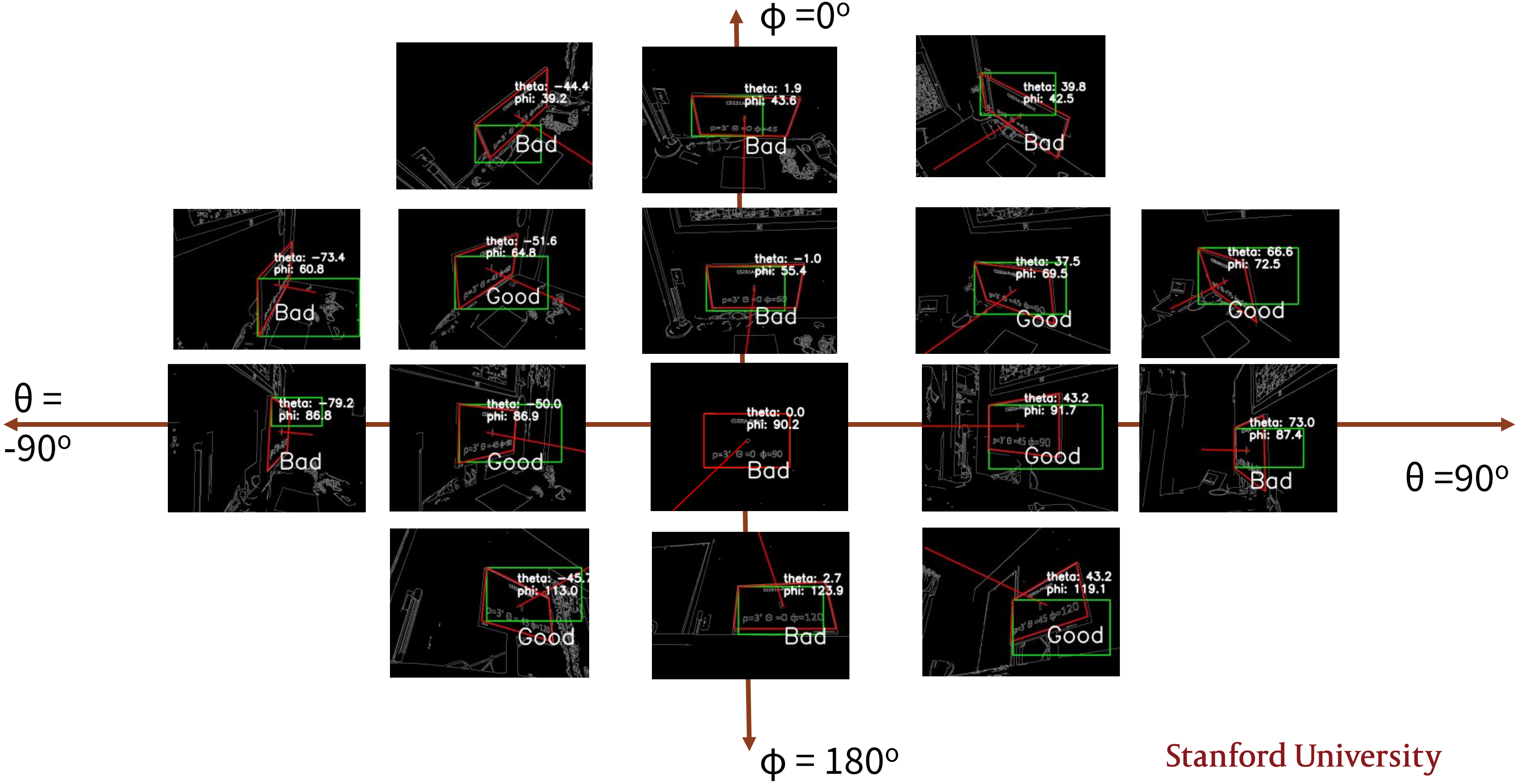
Based on measured values of θ, φ, ρ, a 50-image dataset is built to validate the approach



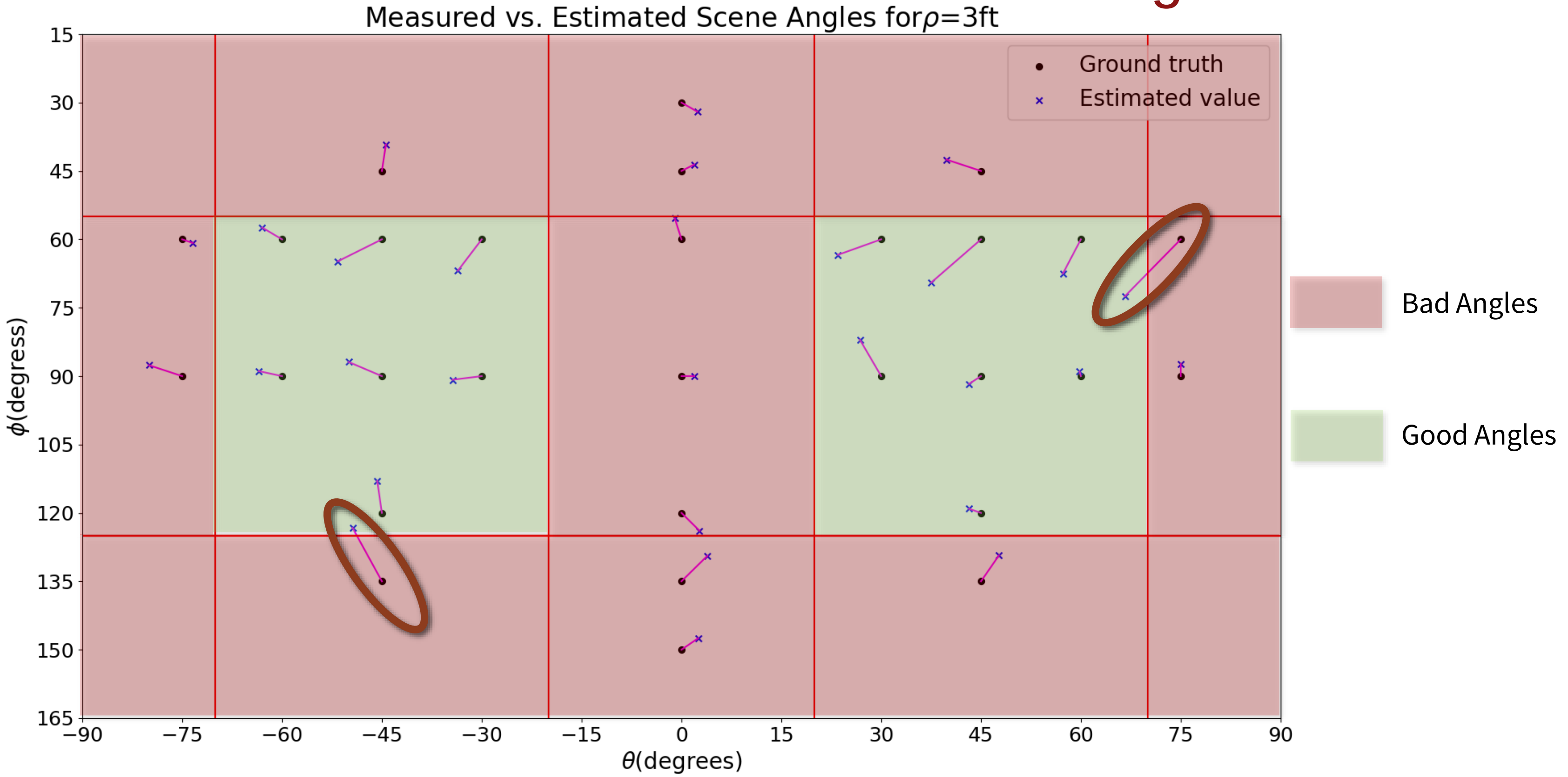
ULDH Sample Angle Estimates/Decisions

Ground truth: θ =-75, ϕ=90

More ULDH Angle Estimation



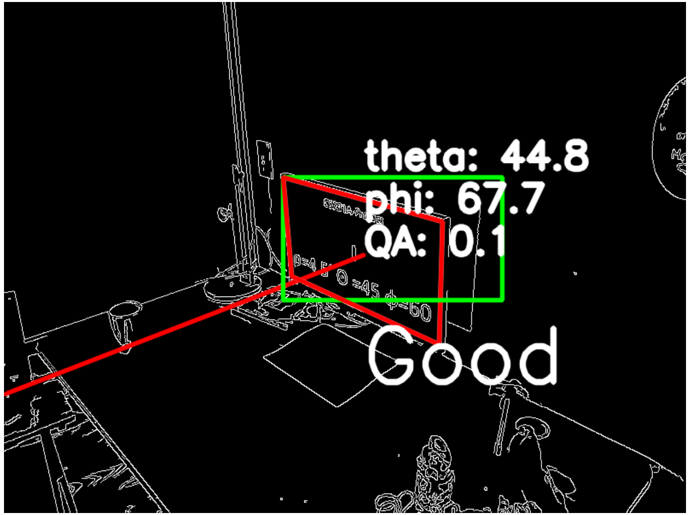
Measured vs. Estimated Polar/Azimuthal Angles



Overall Statistics

Rho	Theta	Phi	Theta_hat	Phi_hat	Theta_delta	Phi_delta	Abs. theta error	Abs Phi error	GT -decision	EST. Decision	Match?
3	30	60	23.5	63.4	6.5	-3.4	6.5	3.4	Good	Good	Match
3	45	60	37.5	69.5	7.5	-9.5	7.5	9.5	Good	Good	Match
3	75	60	66.6	72.5	8.4	-12.5	8.4	12.5	Bad	Good	Mismatch
3	-75	60	-73.4	60.8	-1.6	-0.8	1.6	0.8	Bad	Bad	Match
3	-30	60	-33.6	66.9	3.6	-6.9	3.6	6.9	Good	Good	Match
3	-45	60	-51.6	64.8	6.6	-4.8	6.6	4.8	Good	Good	Match
3	0	90	1.9	90	-1.9	0	1.9	0	Bad	Bad	Match
3	45	90	43.2	91.7	1.8	-1.7	1.8	1.7	Good	Good	Match
3	60	90	59.8	89	0.2	1	0.2	1	Good	Good	Match
3	30	90	26.8	82	3.2	8	3.2	8	Good	Good	Match
3	-60	90	-63.5	88.9	3.5	1.1	3.5	1.1	Good	Good	Match
3	-75	90	-80	87.6	5	2.4	5	2.4	Bad	Bad	Match
3	-45	90	-50	86.9	5	3.1	5	3.1	Good	Good	Match
3	75	90	75	87.4	0	2.6	0	2.6	Bad	Bad	Match
3	-30	90	-34.4	90.8	4.4	-0.8	4.4	0.8	Good	Good	Match
3	0	60	-1	55.4	1	4.6	1	4.6	Bad	Bad	Match
3	60	60	57.3	67.5	2.7	-7.5	2.7	7.5	Good	Good	Match
3	-60	60	-63	57.4	3	2.6	3	2.6	Good	Good	Match
3	0	45	1.9	43.6	-1.9	1.4	1.9	1.4	Bad	Bad	Match
3	45	45	39.8	42.5	5.2	2.5	5.2	2.5	Bad	Bad	Match
3	-45	45	-44.4	39.2	-0.6	5.8	0.6	5.8	Bad	Bad	Match
3	0	120	2.7	123.9	-2.7	-3.9	2.7	3.9	Bad	Bad	Match
3	-45	120	-45.7	113	0.7	7	0.7	7	Good	Good	Match
3	45	120	43.2	119.1	1.8	0.9	1.8	0.9	Good	Good	Match
3	-45	135	-49.4	123.3	4.4	11.7	4.4	11.7	Bad	Good	Mismatch
3	0	135	3.9	129.4	-3.9	5.6	3.9	5.6	Bad	Bad	Match
3	45	135	47.7	129.3	-2.7	5.7	2.7	5.7	Bad	Bad	Match
3	0	30	2.4	32	-2.4	-2	2.4	2	Bad	Bad	Match
3	0	150	2.5	147.5	-2.5	2.5	2.5	2.5	Bad	Bad	Match
4.5	0	90	5.9	99.7	-5.9	-9.7	5.9	9.7	Bad	Bad	Match
4.5	45	60	44.8	67.7	0.2	-7.7	0.2	7.7	Good	Good	Match
7.5	30	90	29.5	86	0.5	4	0.5	4		Good	
						MEAN	3.17	4.49			

Distance Heuristics Checks



Ground truth: θ =45, ϕ=60, ρ=4.5’

Ground truth: θ =30, ϕ=90, ρ=7.5’

What about original motivation?

Conclusion

- A robust approach to estimate scene geometry for remote exam with a side camera is proposed
- Key Contributions:
  - A new simple ULDH algorithm to estimate polar/azimuthal angles
  - Complete image pipeline from raw image to decision on camera geometry
  - A 50-image dataset with measured ground truth angle/distance
- Future Work Possibilities
  - Analytically/experimentally further validate ULDH’s efficacy
  - Make Canny+Hough+Quad-detector faster/robust with CNN object detectors

Questions?