Rigid-body Dynamics for Articulated Mesh Tracking

Leonid Keselman (Intel)

Sterling Orsten (Intel)

Stan Melax (formerly @ Intel)

Intel Legal Disclaimer

<Demo>

- Run on my IT-issued laptop
  - i5-4300U @ 1.90 GHz
  - This is the machine used for any demos/timings in talk
  - Algorithm code is all single-core C++ CPU code
    - By choice

Technical Reference Materials

- Physics Engine
  - https://github.com/melax/sandbox
  - Stan Melax's sandbox for physics + graphics code, BSD license
  - Sequential Iterative Impulse solver, © 1998-2008
    - Our tracking built on top of improved/expanded version

- Intel's 2013 release of this work, free download
  - https://software.intel.com/en-us/articles/the-intel-skeletal-hand-tracking-library-experimental-release
  - Camera input layer is sample code, you could re-purpose on top of whatever data you'd like

- Demo Videos & Concepts
  - https://www.youtube.com/user/smelax

Academic Reference Material

- Melax, Keselman, Orsten.
  - *Dynamics based 3D skeletal hand tracking*.
  - i3D 2013. Poster
  - GI 2013, Full-length paper

Talk Overview

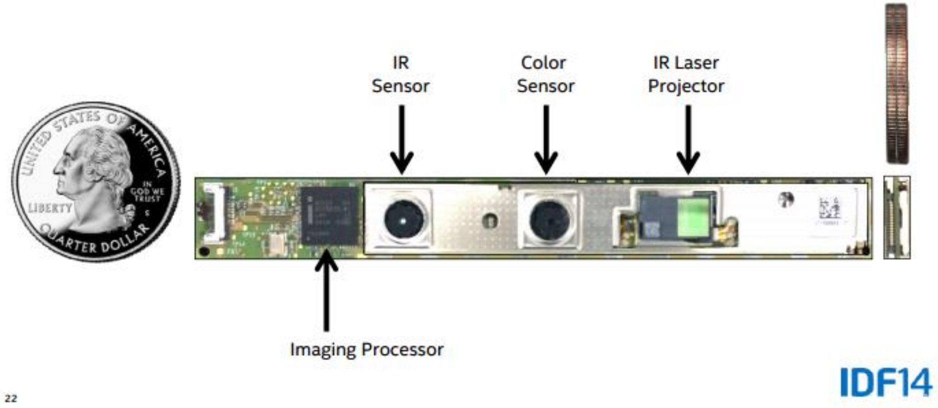
1. Motivation
2. Dynamics-Based Tracking
    a) Background
    b) Method overview
    c) Hand Model
3. Fast iterative Tracking
    a) Our tracking architecture
    b) Benefits of being 3D
    c) Multi-hypothesis architecture
    d) Value of working in a constraint-based solver
4. Cameras and Usages
    a) Basic Filter Architecture
    b) Structured Light: PrimeSense & Kinect v1
    c) Time-of-Flight: SoftKinetic
    d) Projected Texture Stereo: Intel R200
    e) Structured Light: Intel F200
5. Annotation, Learning and Classification
6. Q&A

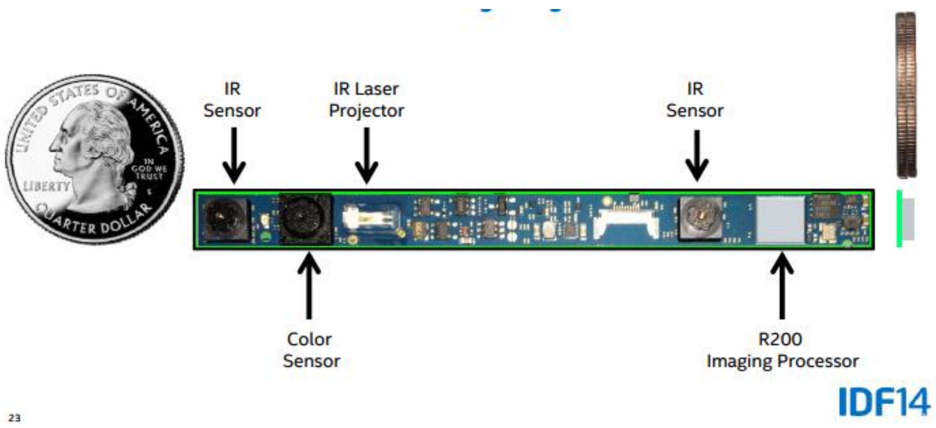
Background

Background

- Intel interested in depth cameras
  - Started in ~2011
  - Most of our work was during 2012
- January 2013: “Senz 3D”
  - QVGA, TOF depth camera
  - CES 2013 Launch
- Present Day: Intel RealSense
  - F200 & R200

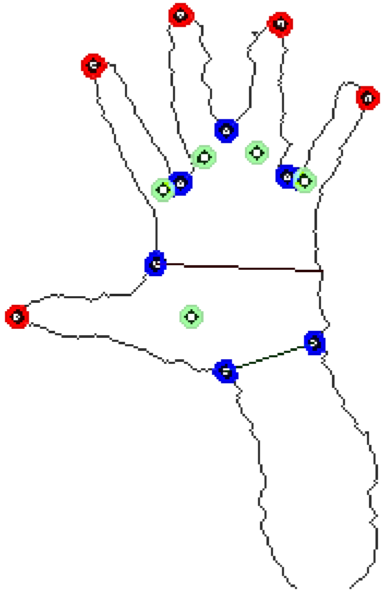
Background

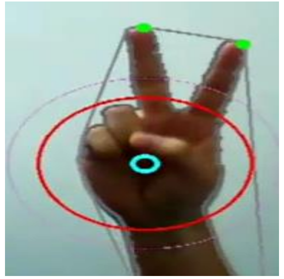


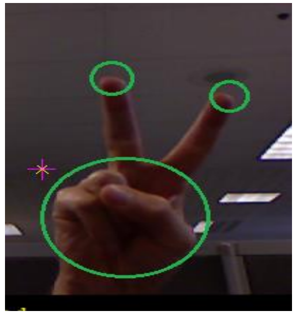


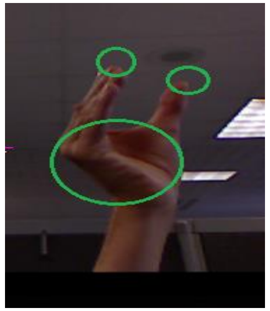
- Intel has 2 depth sensors available as developer kits
- http://click.intel.com/realsense.html
- F200
  - Structured Light
- R200
  - Projected Texture Stereo

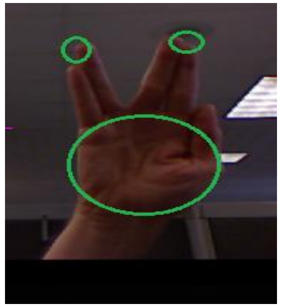
2011: Real-time “Hand Tracking” from 3D cameras



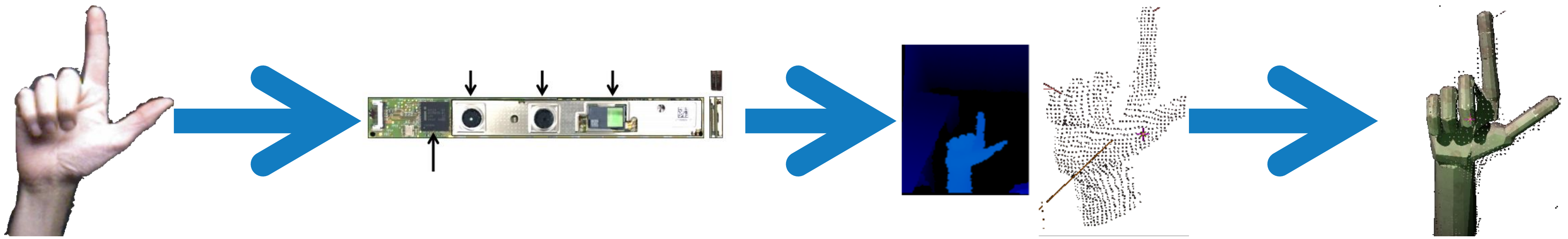








3D Hand Tracking Goal



- Full 6 DOF pose for all finger bodies
  - Along with sufficient information to provide collisions and interactions
  - On consumer hardware

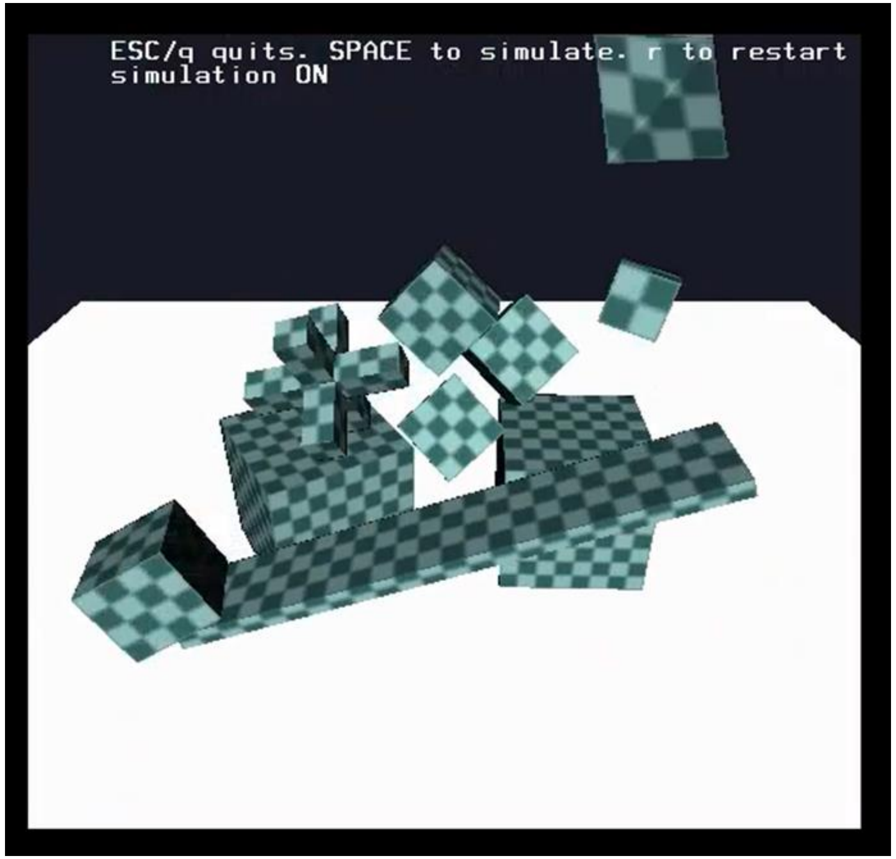
- Existing work on providing such 3D pose
  - Wang, Popovic. Real-time hand-tracking with a color glove. ‘09. + 6D Hands Pose Template
  - Hilliges et al. Digits: freehand 3D interactions anywhere using a wrist-worn gloveless sensor UIST ‘12.
  - **Oikonomidis, Kyriazis, Argyros. Efficient model-based 3D tracking of hand articulations using Kinect. BMVC ‘11.**

Motivation: Emergent Interaction

Dynamics-based Tracking

Rigid Body Dynamics

- Ability to physically simulate articulated models with collisions and joints.
- Achieved by satisfying linear and angular constraints.

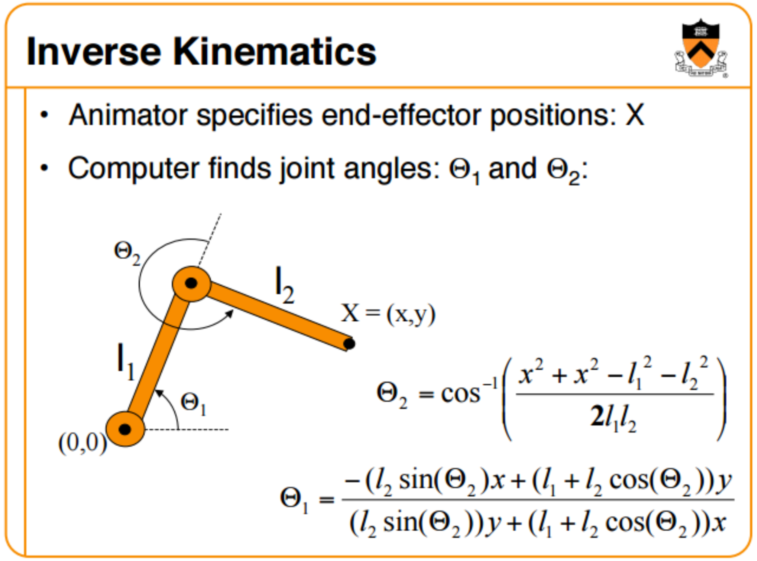


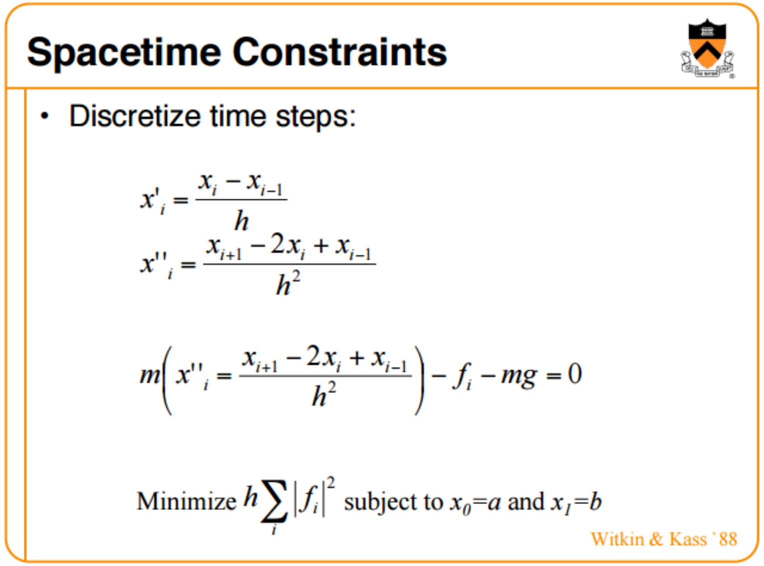
Rigid Body Dynamics

- Ability to physically simulate articulated models with collisions and joints.
- Achieved by satisfying linear and angular constraints.

Kinematics vs Dynamics

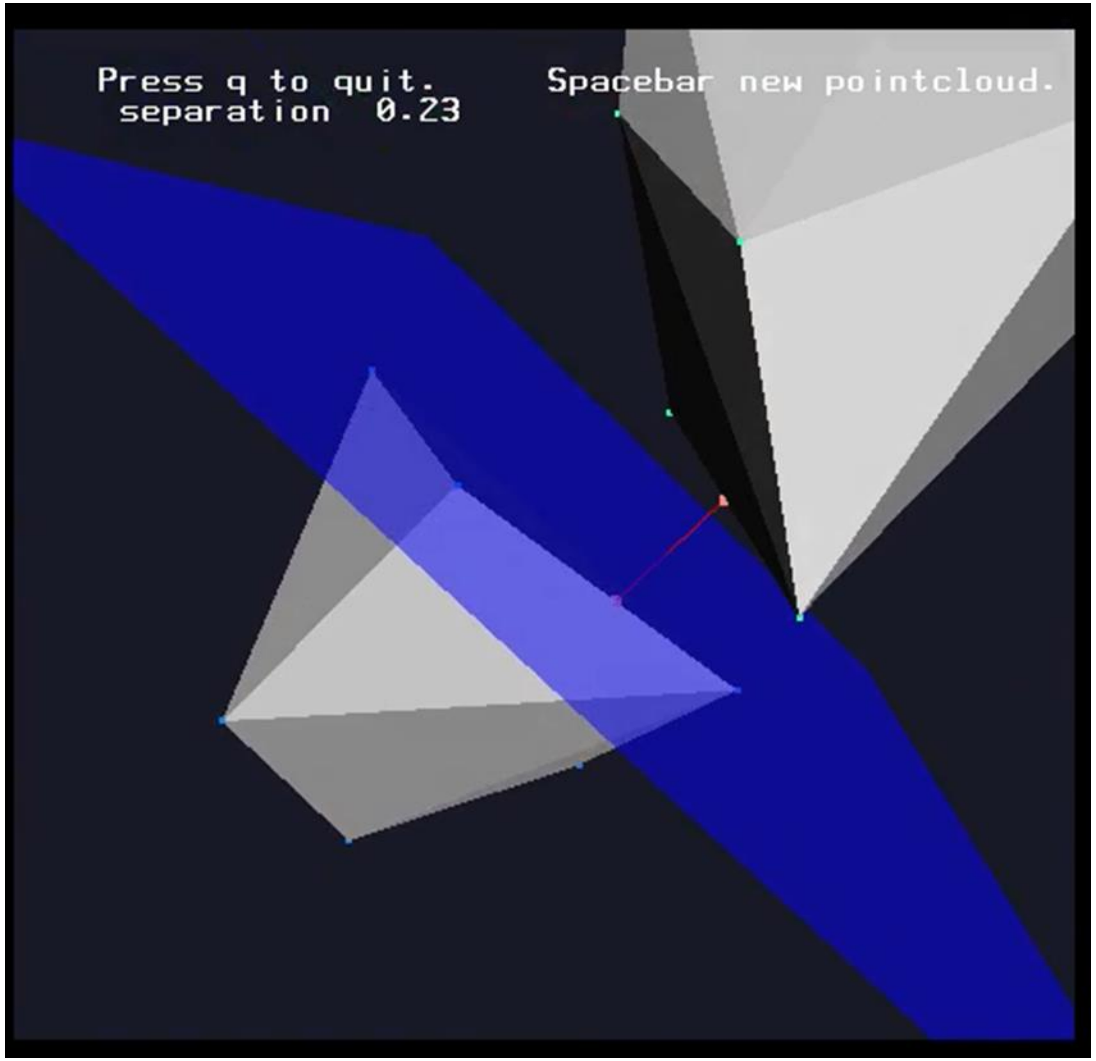
- From Adam Finkelstien's COS426 lecture notes:

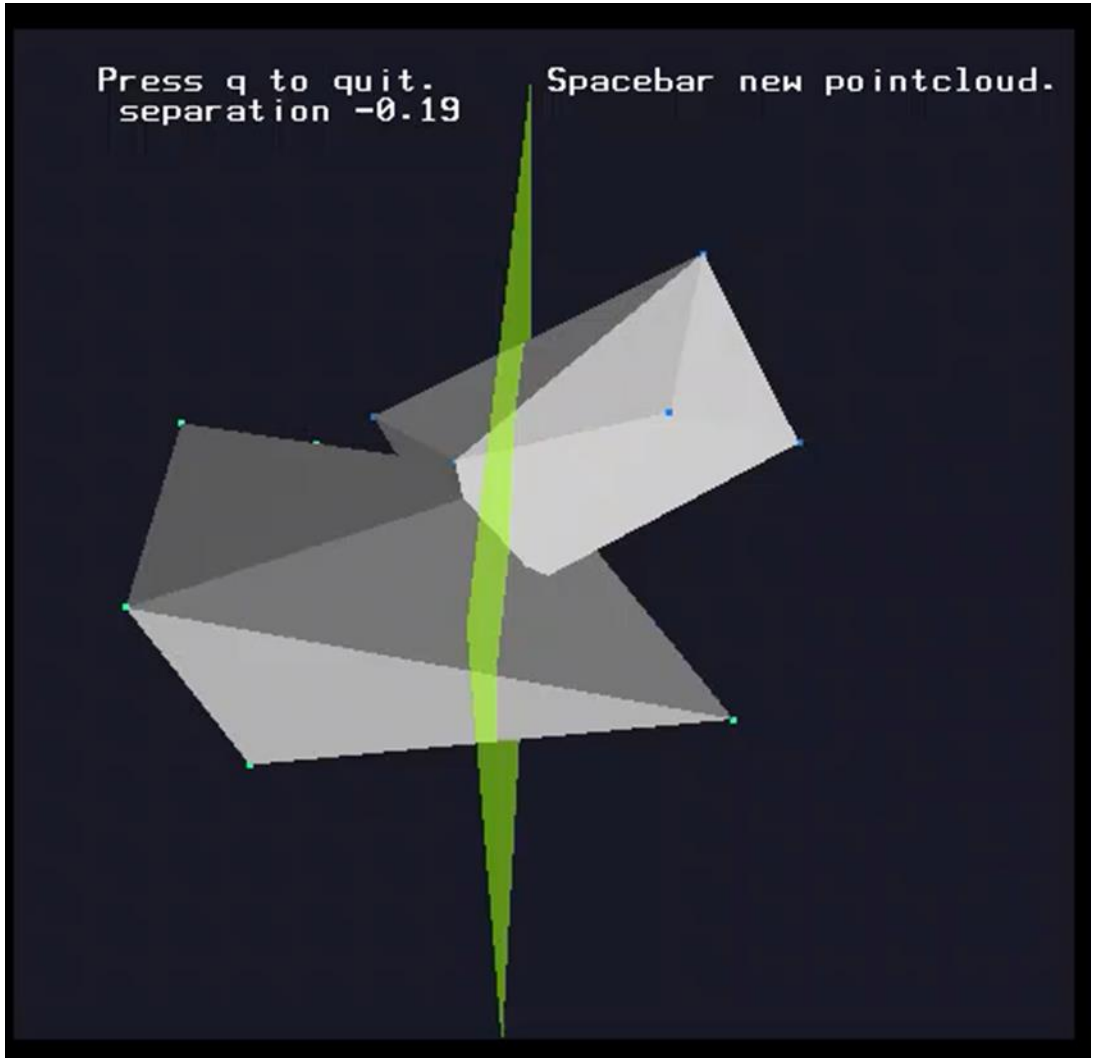




- Kinematics
  - Considers only motion
  - Determined by positions, velocities, accelerations
- Dynamics
  - Considers underlying forces
  - Compute motion from initial conditions and physics

GJK 1988

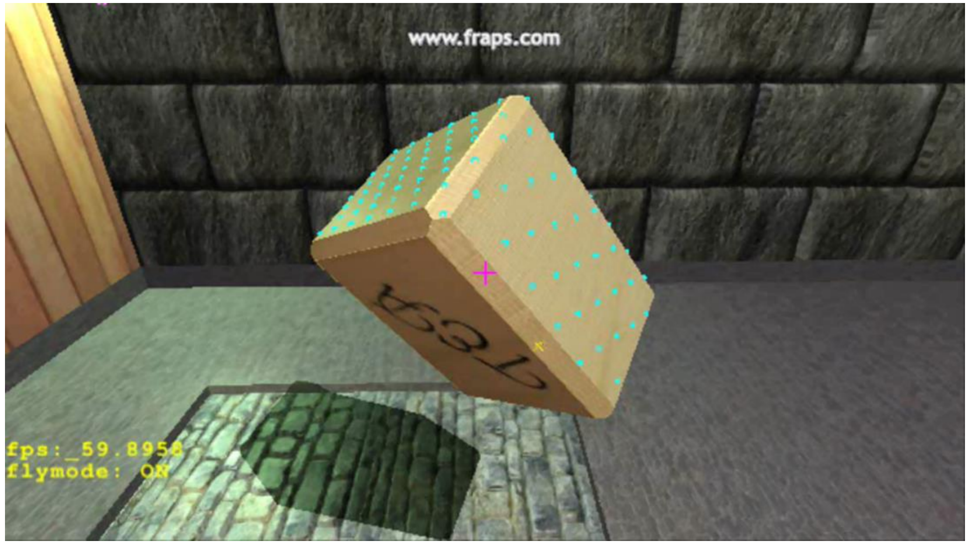

Press q to quit.
separation 0.23
Spacebar new pointcloud.


Press q to quit.
separation -0.19
Spacebar new pointcloud.

Surface constraints

- Like little magnets that attract the surface

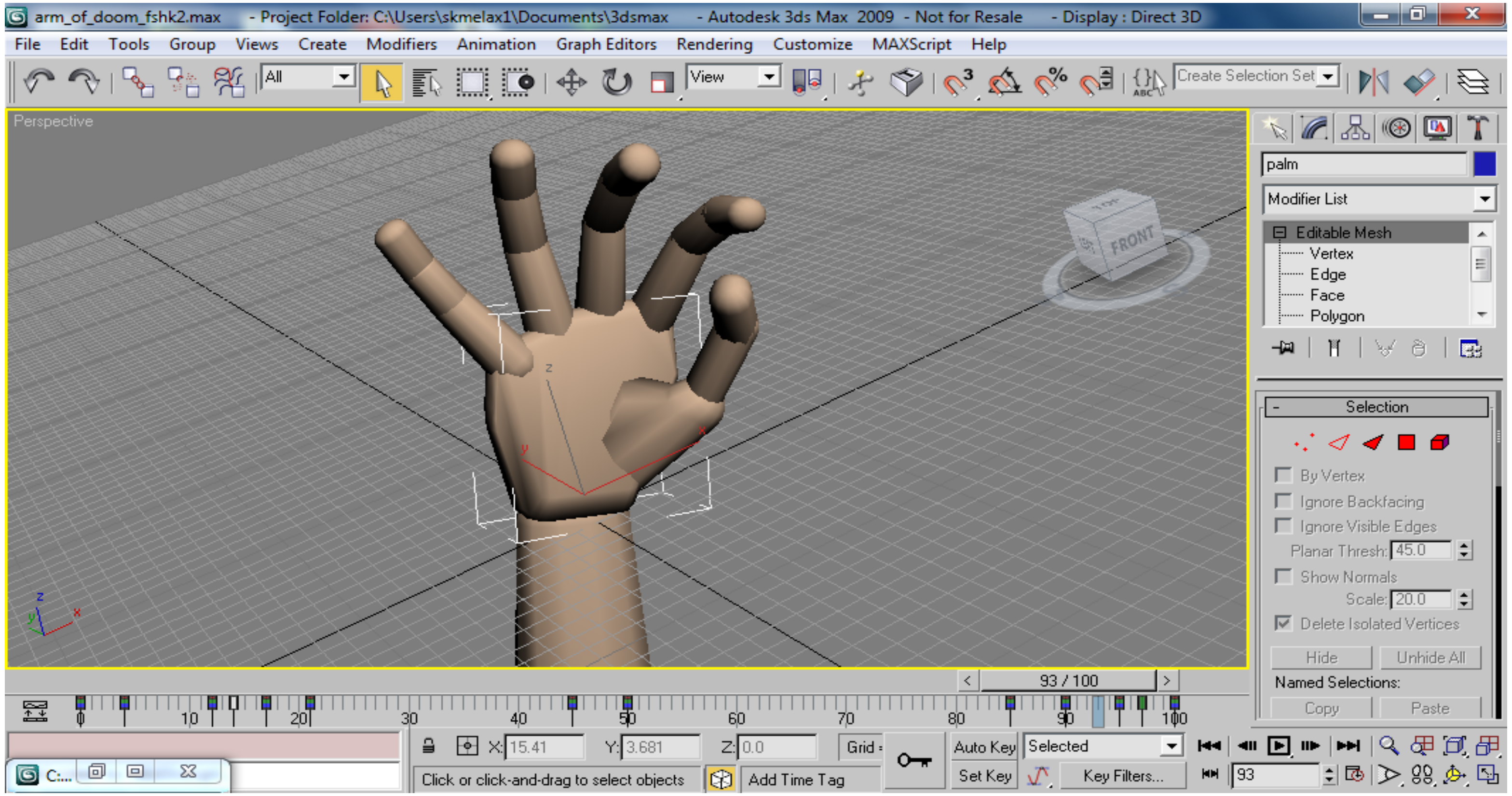
Surface constraints created from synthetically generated depth data

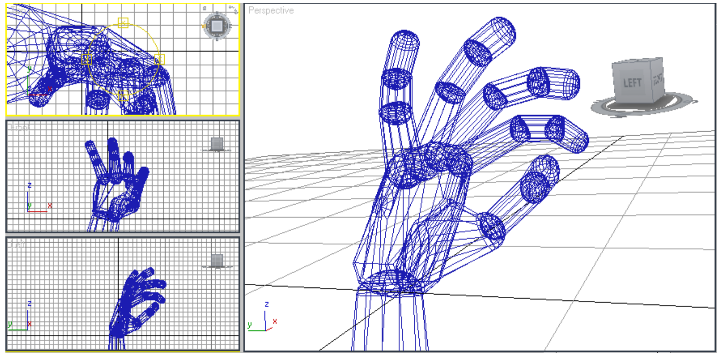


2nd instance (left), generates depth data (middle), tracking/fitting (right)

Model to track authored in 3DS max

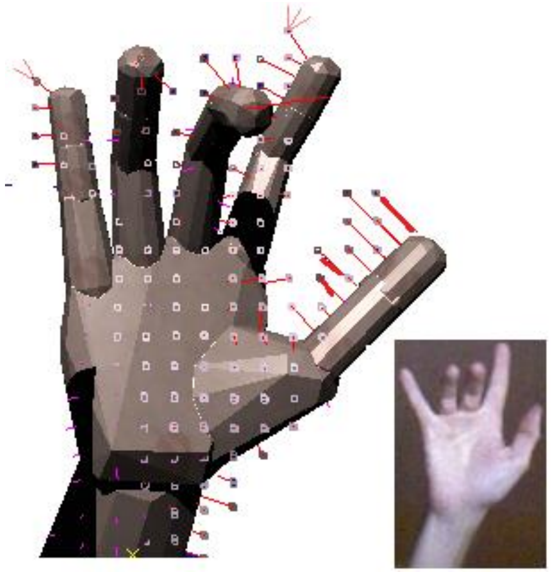
- Created a generic hand and scale it as necessary:

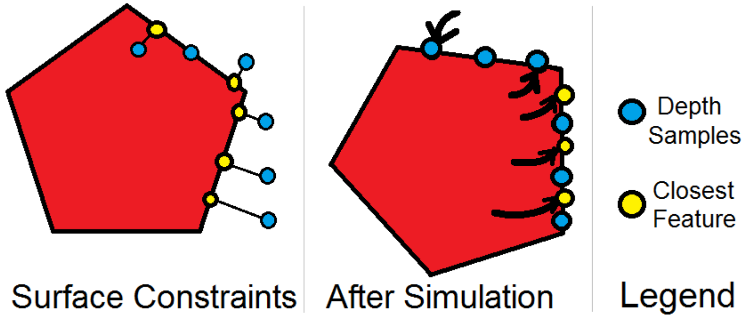


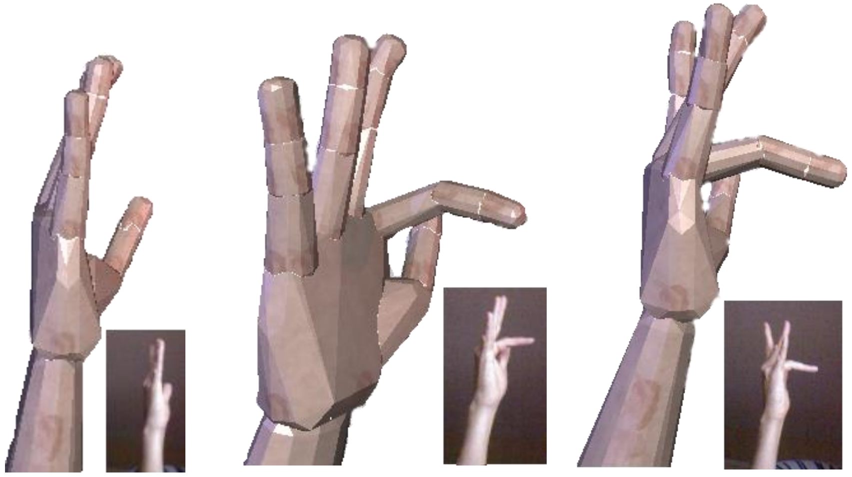


Custom models could be made.

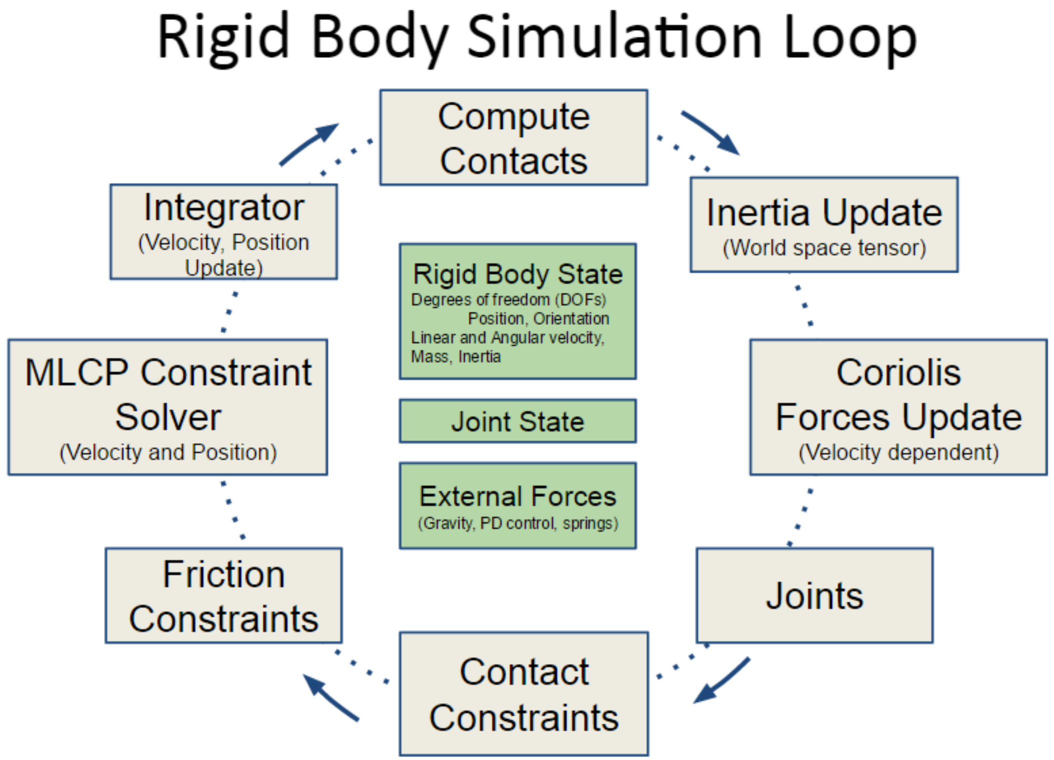
Combining constraints + hand mesh







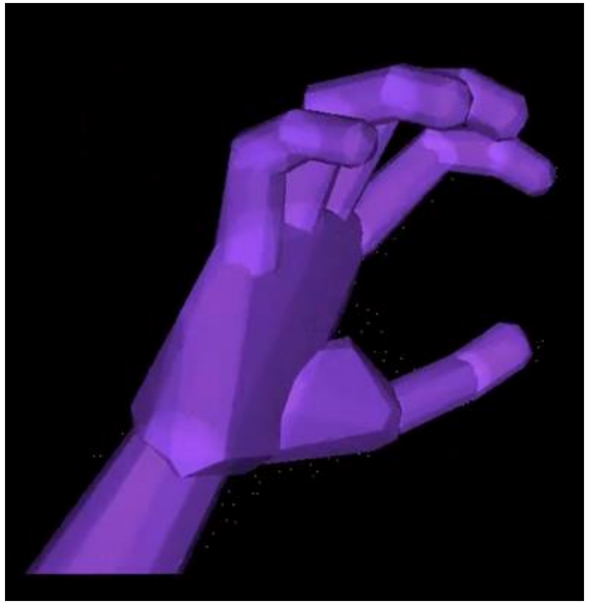
Rigid Body Dynamics



- Picture from E. Coumans' talk GDC14 on MLCP solvers
  - http://goo.gl/84N71q
- Many methods
  - Stable, Approximate
  - Minimal tuning, temporally consistent
  - Very, very fast
  - (30-1000Hz for modern games)
- We use a sequential impulse solver.
  - Fast, stable, converges to global solution
  - See reference slides for more details

Rigid Body Dynamics: Easy to reason with

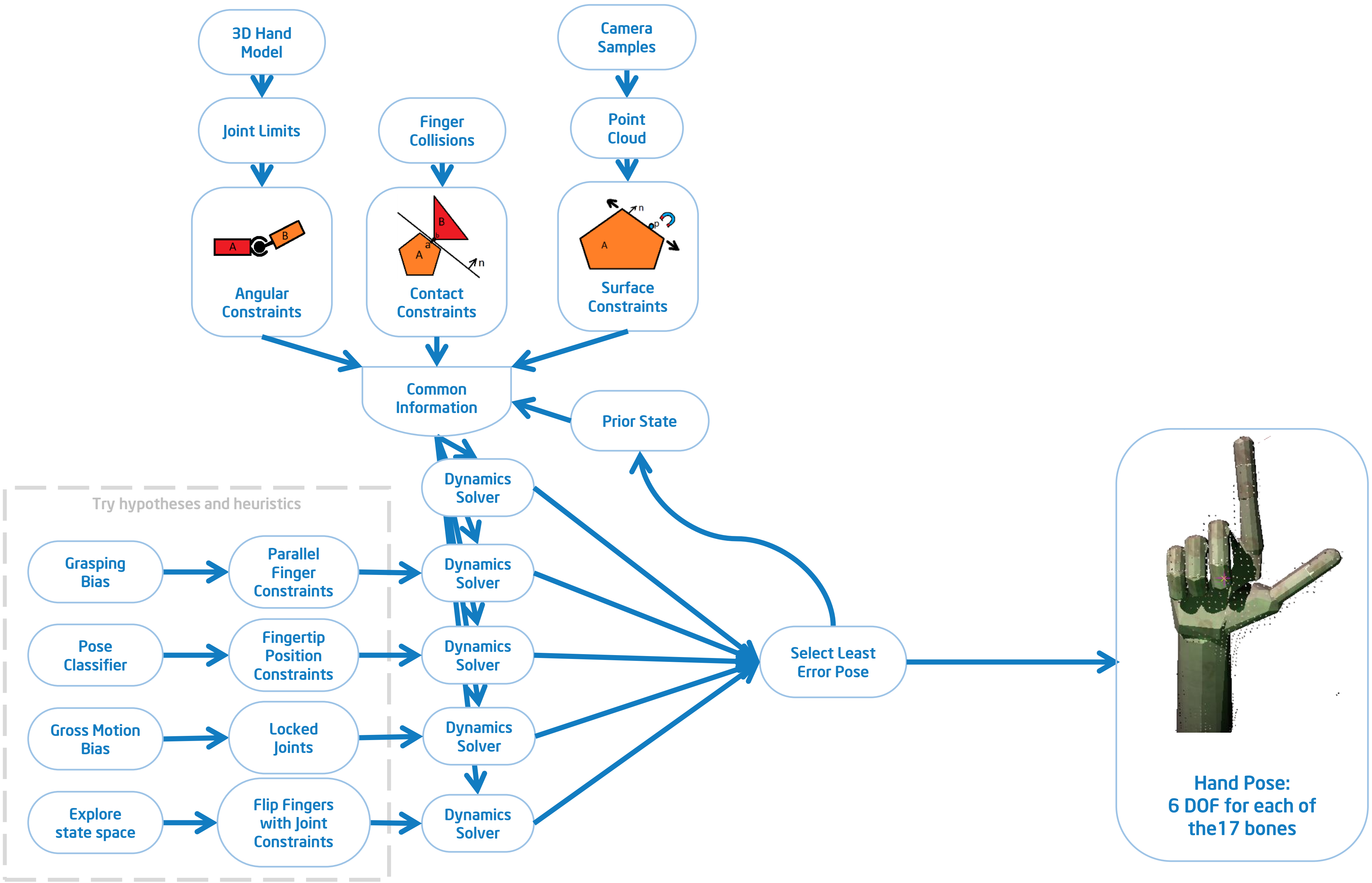
Unconstrained



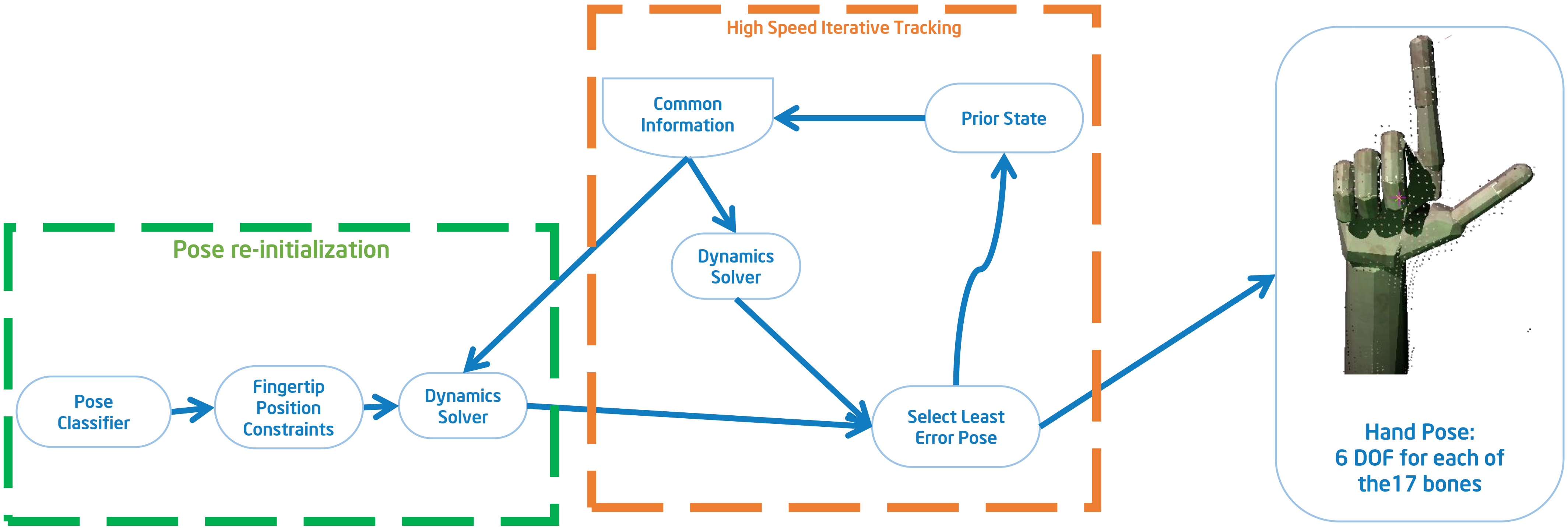
With bend angle constraint

- The use of a single unified solver
  - Collisions
  - Angular limits
  - Data to model minimization
  - Approximation to real-world
- Solves an MLCP: an arbitrary set of angular and linear constraints
- Easy to express new information into the system
  - Force fingertip to bend at expected relative angle?
  - Just add a conical constraint!

Fast Iterative Tracking


3D Hand Model
Joint Limits
Finger Collisions
Camera Samples
Point Cloud
A
B
Angular Constraints
B
A
a
b
n
Contact Constraints
n
p
A
Surface Constraints
Common Information
Prior State
Try hypotheses and heuristics
Grasping Bias
Parallel Finger Constraints
Pose Classifier
Fingertip Position Constraints
Gross Motion Bias
Locked Joints
Explore state space
Flip Fingers with Joint Constraints
Dynamics Solver
Dynamics Solver
Dynamics Solver
Dynamics Solver
Dynamics Solver
Select Least Error Pose
Hand Pose:
6 DOF for each of
the17 bones

Simplified Architecture view



This type of track + reinitialize architecture seems to be catching on

- CVPR 14: Qian et al, “Realtime and Robust Hand Tracking from Depth”
- CHI 15: Sharp et al, "Accurate, Robust, and Flexible Real-time Hand Tracking."

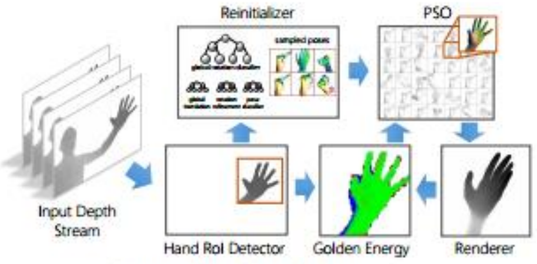
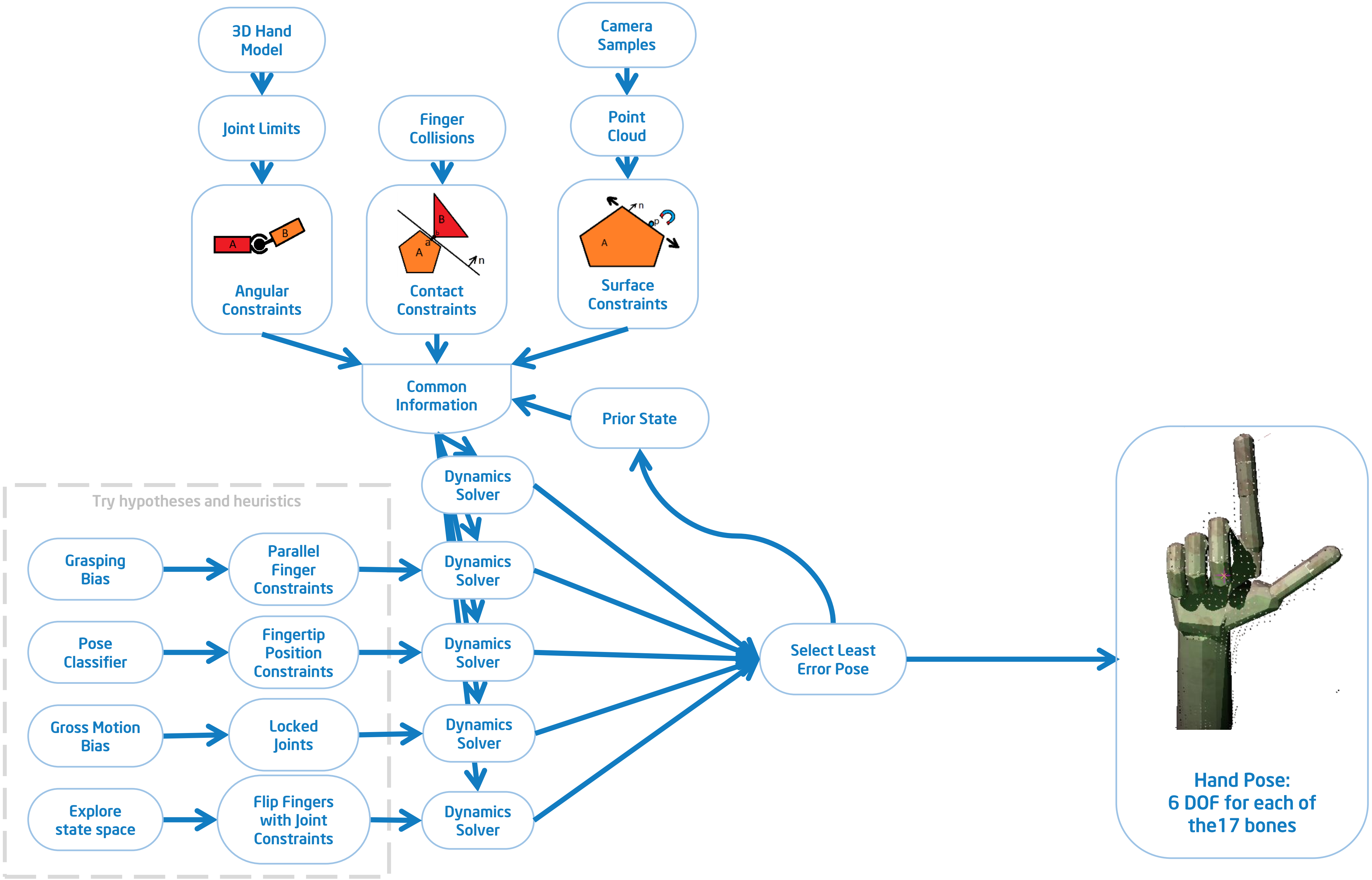
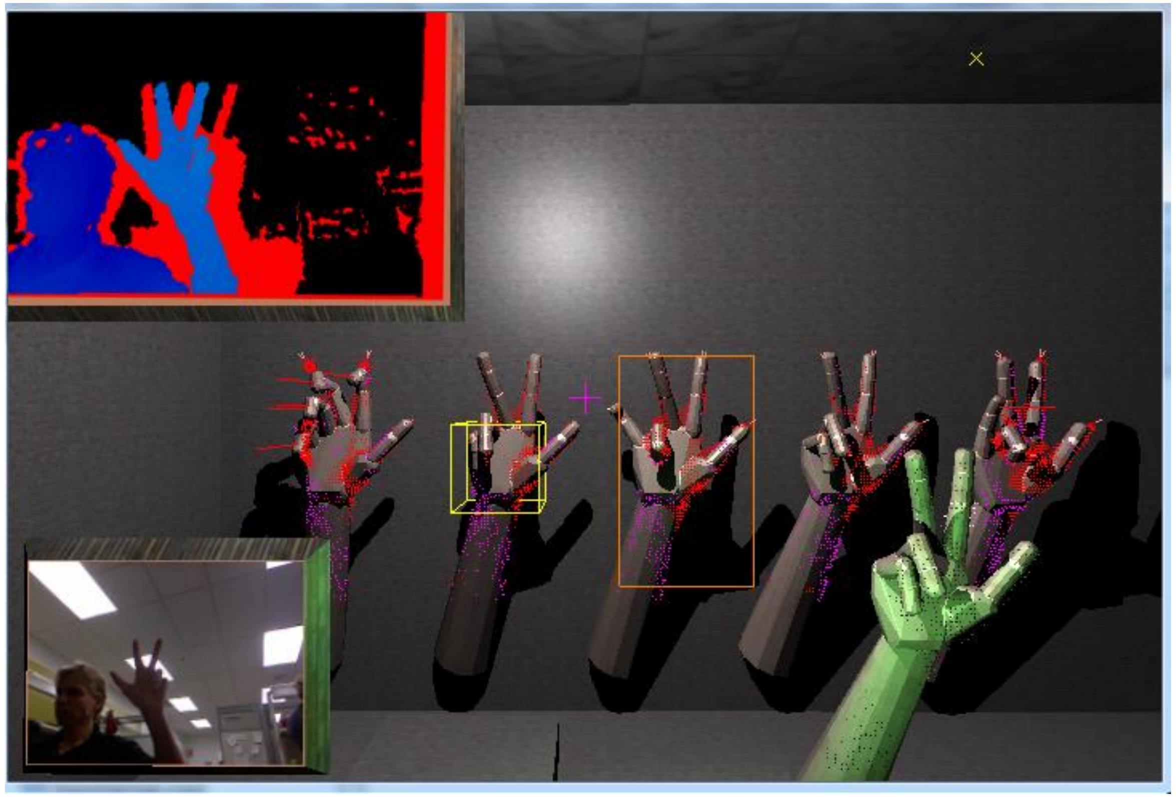


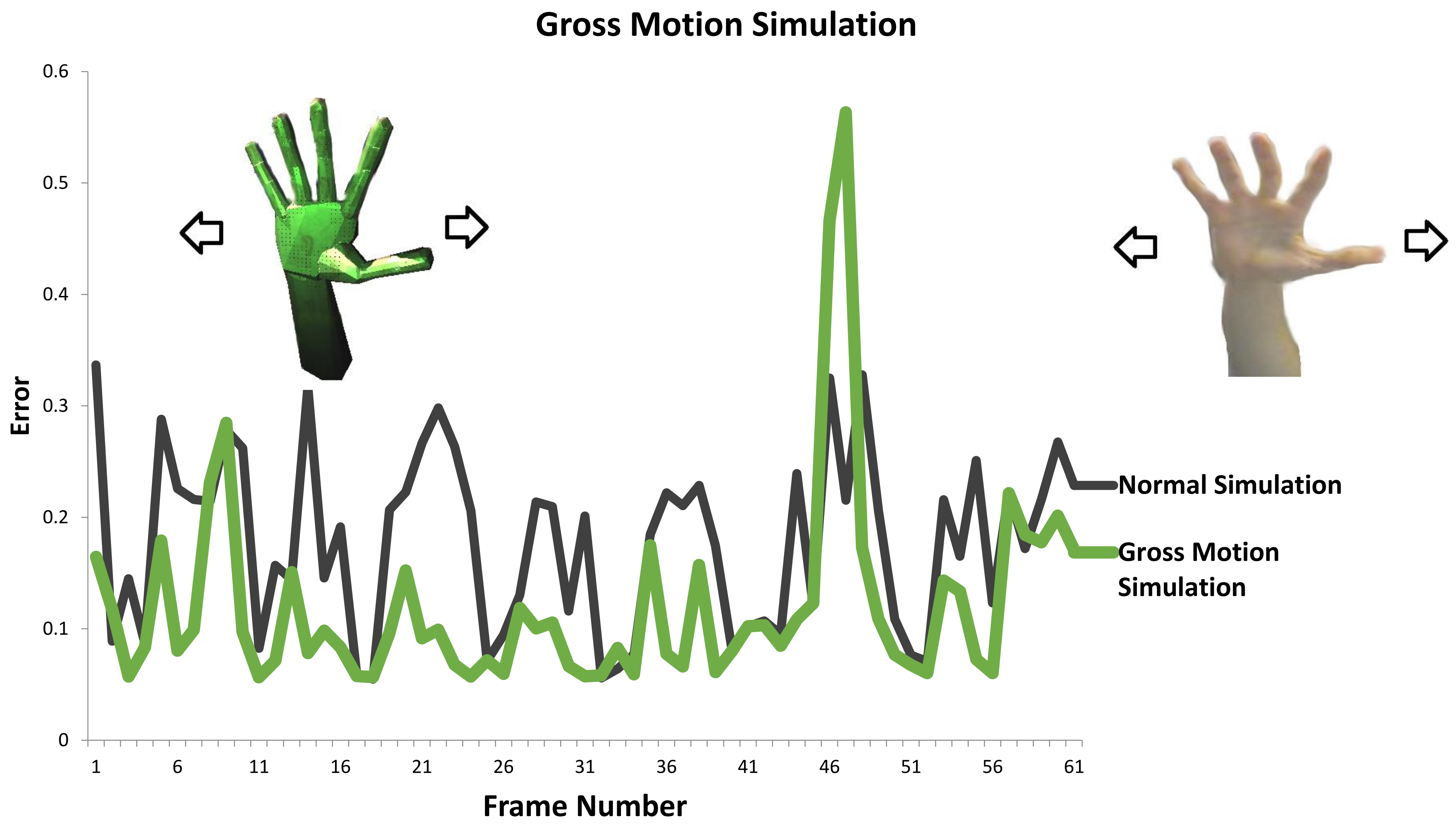
Figure 2: Algorithm pipeline.

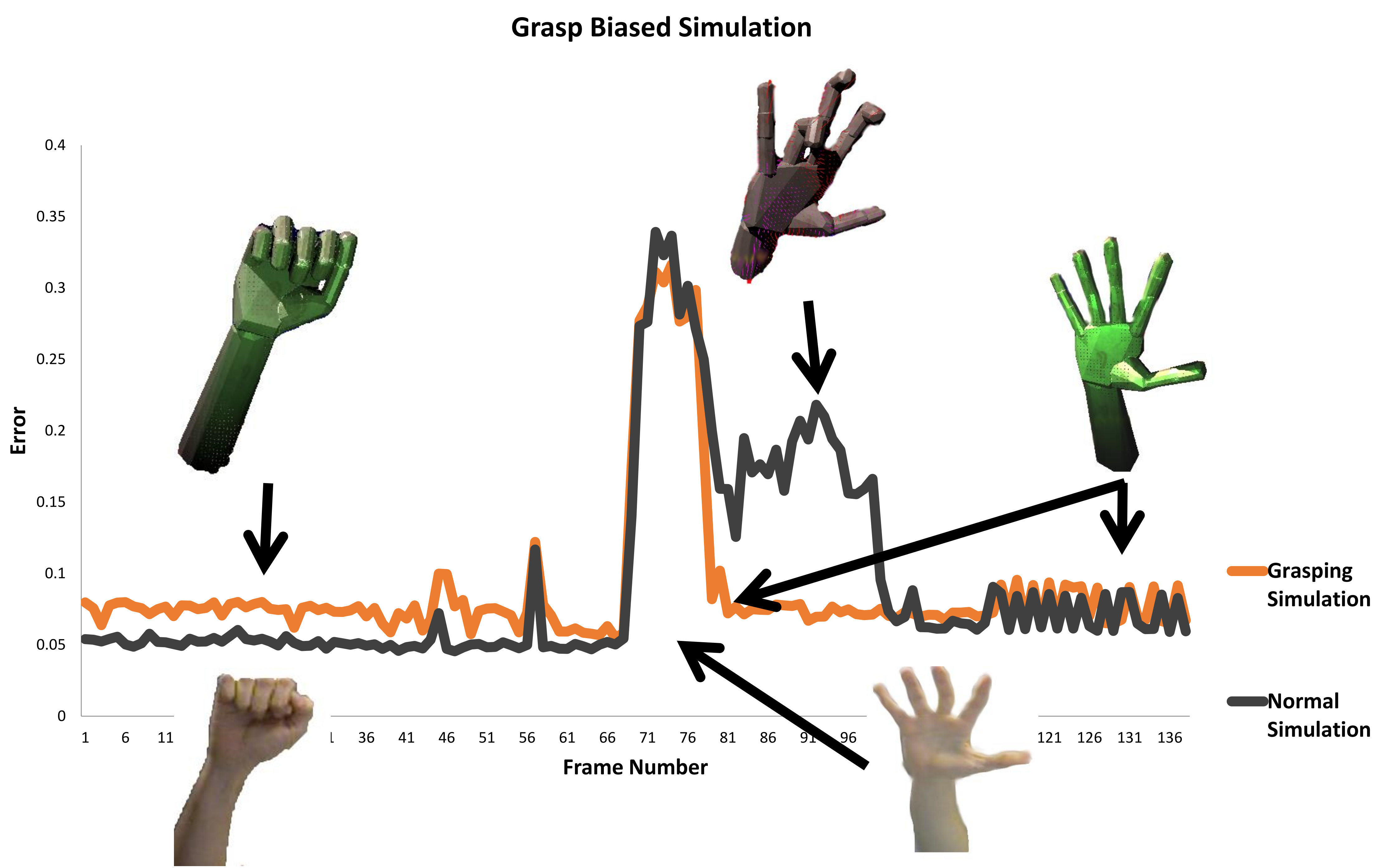

3D Hand Model
Joint Limits
Finger Collisions
Camera Samples
Point Cloud
Angular Constraints
Contact Constraints
Surface Constraints
Common Information
Prior State
Dynamics Solver
Dynamics Solver
Dynamics Solver
Dynamics Solver
Dynamics Solver
Try hypotheses and heuristics
Grasping Bias
Parallel Finger Constraints
Pose Classifier
Fingertip Position Constraints
Gross Motion Bias
Locked Joints
Explore state space
Flip Fingers with Joint Constraints
Select Least Error Pose
Hand Pose:
6 DOF for each of the17 bones

Multiple Simulations

- System can get stuck in local minimum
- Run multiple simulations and pick the best fit.
- Increase likelihood of regaining lost tracking

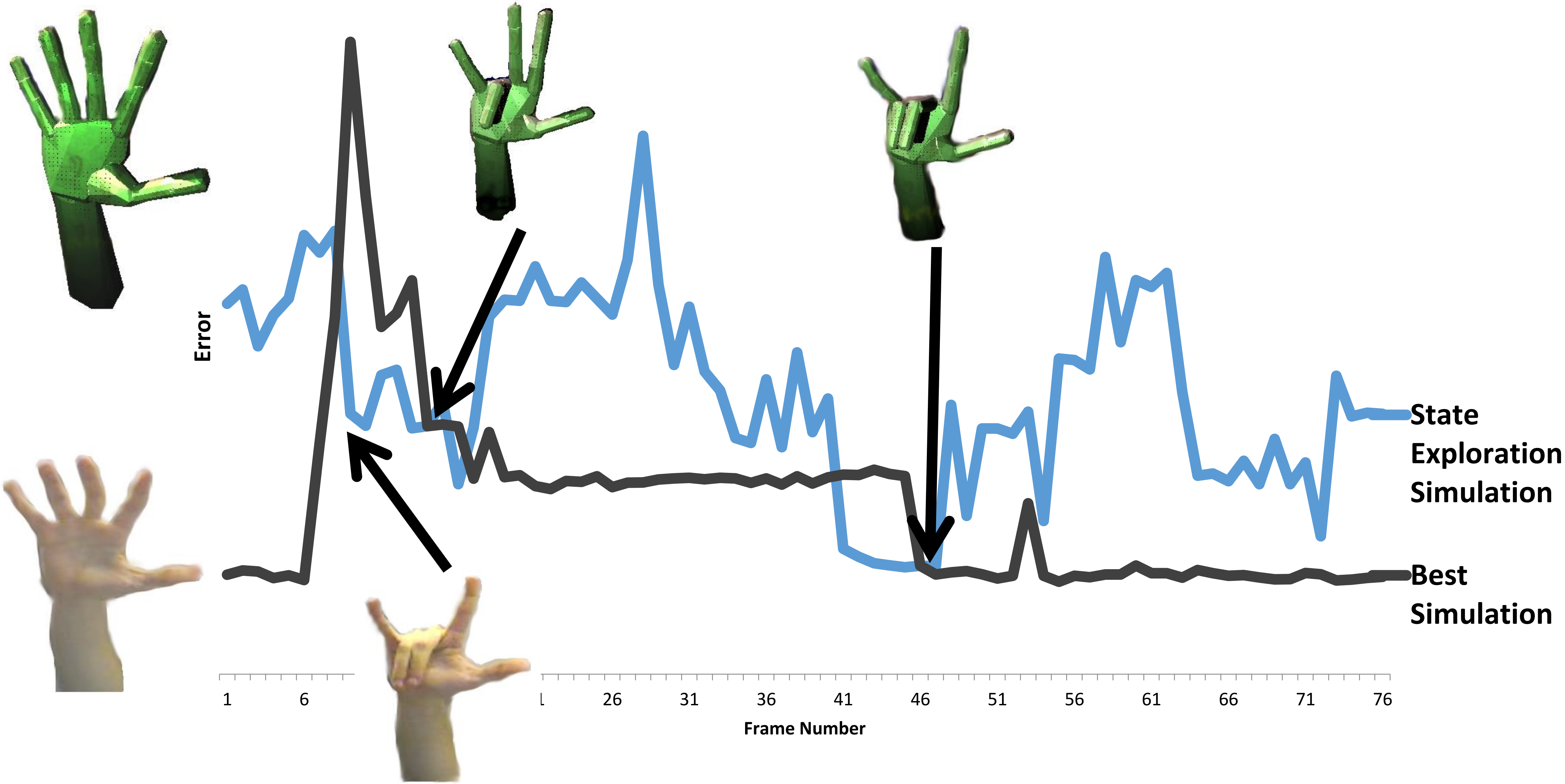



Gross Motion Simulation
0.6
0.5
0.4
0.3
0.2
0.1
0
Error
1
6
11
16
21
26
31
36
41
46
51
56
61
Frame Number
Normal Simulation
Gross Motion Simulation



Grasp Biased Simulation
Error
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
1
6
11
36
41
46
51
56
61
66
71
76
81
86
96
121
126
131
136
Frame Number
Grasping Simulation
Normal Simulation

State Exploration Simulation



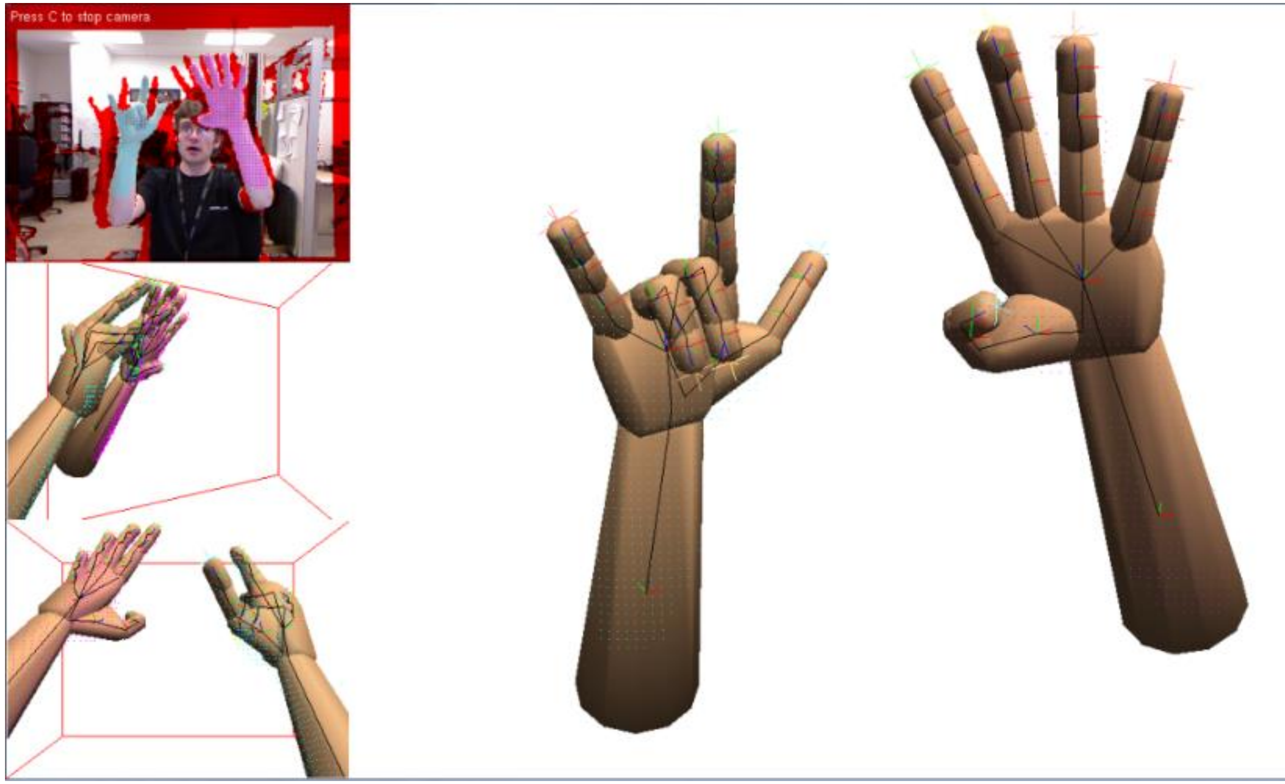
Error Metric

- $E_{model} = \alpha(\|E_{data}\|_{\infty} + E_{occluding})$

- L-inf norm for point cloud to rigid body surface
  - We want the pose that best explains all points
- Additional penalty for bone centroid existing in front of background
  - Single raycast per bone (17 total per frame)
  - Can strengthen penalty to also penalize missing data for ToF cameras
- Non-standard simulations have a penalty multiplier
- Other metrics available, but generative + reprojection based metrics much more computationally expensive

Two Handed Interaction

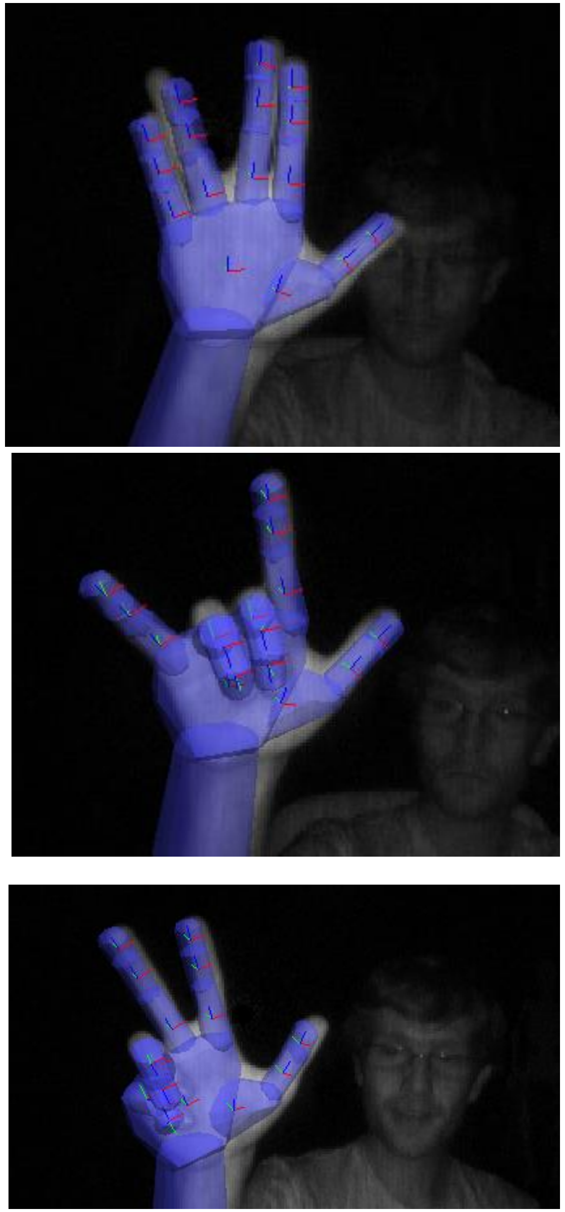
- Simple Method
  - K-Means Merging for Segmentation
  - K = 2
  - Explicitly seed leftmost & rightmost points
  - Merge clusters if centroids are close
  - Run two simulations for 2 hands
- Easy Extension
  - Solve in single simulation
  - Might require more careful correspondence



High speed motion tracking

Results

- Tracked hand model compared to input



Creative Gesture Camera

Asus xtion

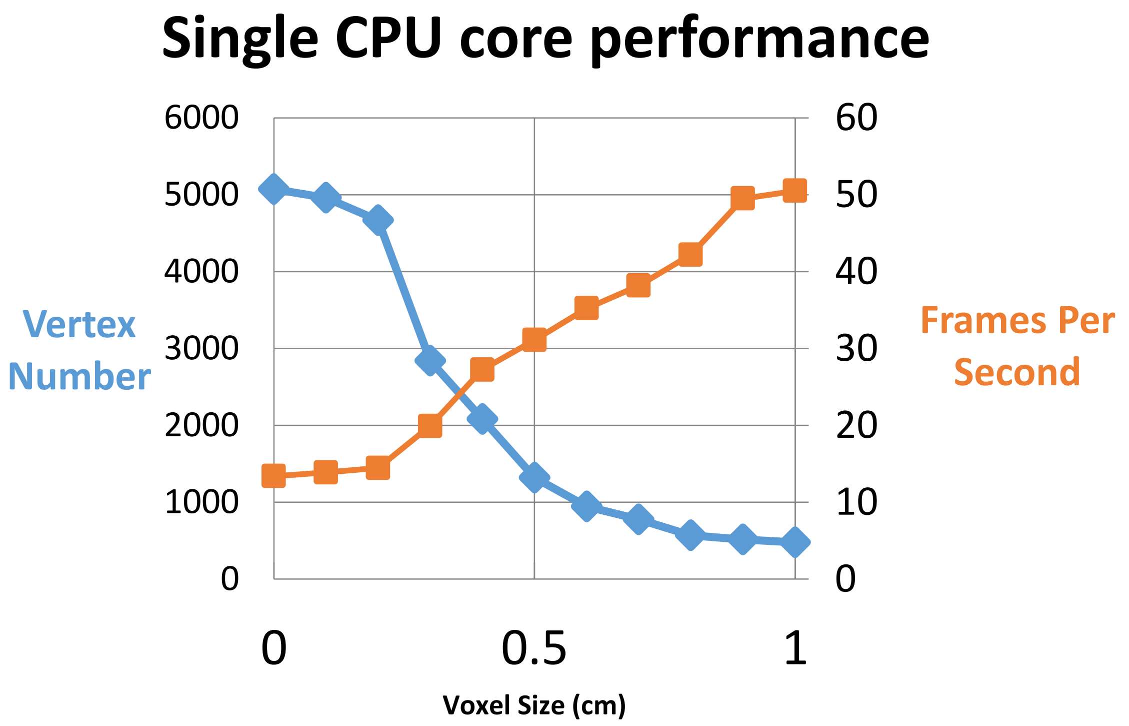
Voxel Subsampling

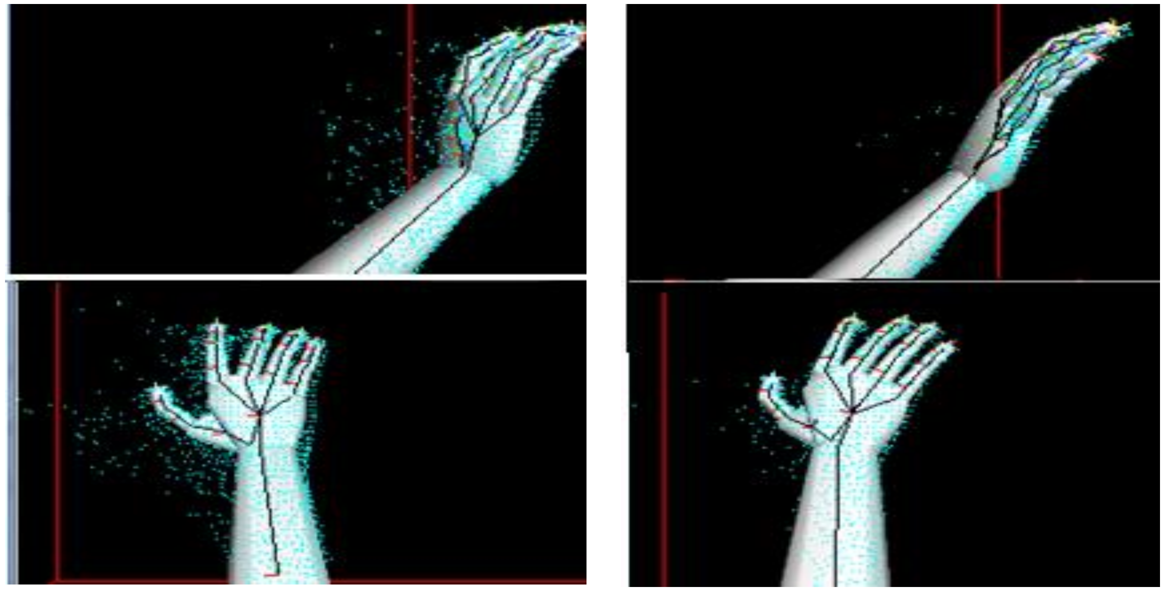
- Would be too expensive to use every depth sample.

- High performance
  - 45-80 FPS on single core
  - Flexible subsampling options
  - Approximate hashing scheme

- Added benefit of removing outliers or “flying pixels”.
  - Configurable density check
- Improves fitting of tracking model.

- For noisy cameras, we also have a custom 16bit spatial median filter and a bilateral filter for photometric-aligned data streams





No Subsampling With Subsampling

Solver Performance

- We're processing roughly ~300-400 volumetric contacts
- Overall, roughly ~5,000 to 6,500 constraints solved per frame in multiple hypothesis solver architecture.
  - Multiple solvers, multiple passes
- This is roughly 50,000 constraint-iterations (~10 iterations per step)
- Total system clock on my i5-4300U is about 6-7 milliseconds
  - ~ 1uS/constraint
  - ~ 0.15uS/constraint-iteration.
- In single hypothesis version, full pass of dense data fitting is ~800uS
- Includes everything after voxel subsample through solver completion, counting
  - Closest surface finding
  - Solving data constraints
  - Solving self-collision constraints
  - Evaluating all error metrics

Benefits of fast iterative tracking

- Fewer points run faster can be a lot more robust.
  - Also computationally more efficient: Relative to velocity, 2D image search space is quadratic with resolution increases, but linear with time decreases.

- If you run fast enough, all changes are small
  - KinectFusion, Newcombe et al, 2011; Lucas & Kanade, 1981
  - *"Real-Time Camera Tracking: When is High Frame-Rate Best?",* Ankur Handa, Richard A. Newcombe, Adrien Angeli, and Andrew J. Davison, ECCV 2012
    - Develops Pareto wavefront for tracking cameras given compute budget. Lower resolution with higher framerate performs better than higher resolution at lower frame-rate

Benefits of fast iterative tracking

- Feel free to throw out whatever data might be noisy
  - Make system robust by being selective
- Can track in extremely sparse data environments
  - passive stereo or in depth camera saturation conditions (only edge data)

Benefits of fast iterative tracking

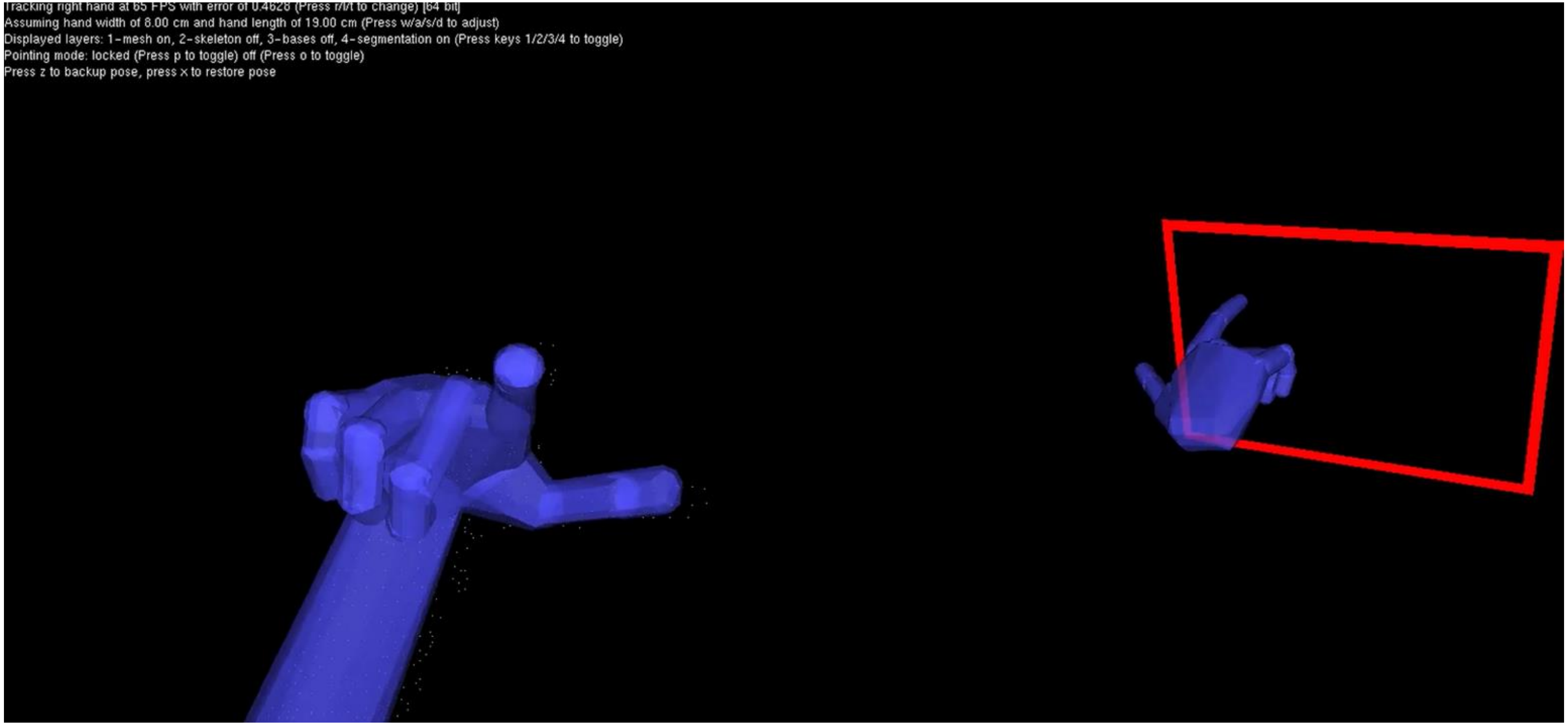
- Robust tracking with minimal data
  - Tracking under camera saturation conditions
  - Temporal Coherence
- Top Right = Input Depth
  - White = no data
  - Gray = depth data
- Bottom Left = Estimated Hand Pose

Cameras and Usages

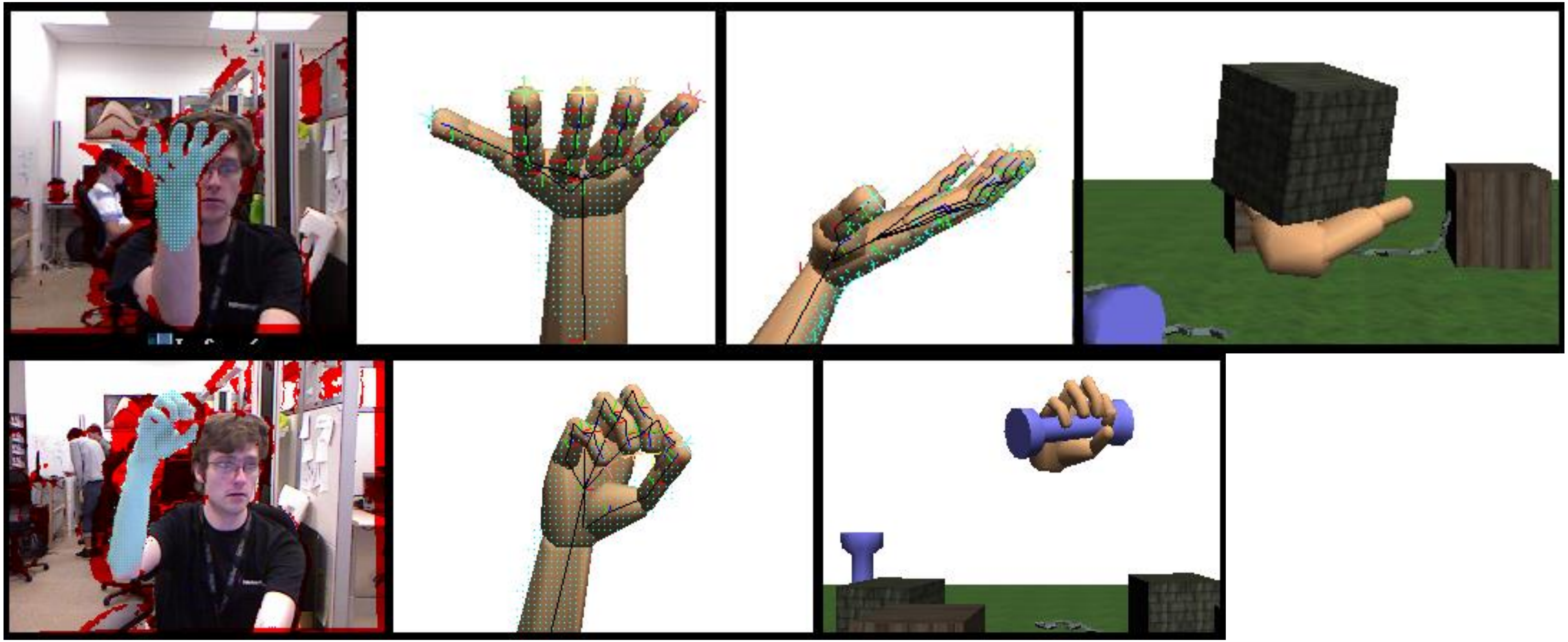
Pose Locking

- Trivial to force the solver to use reduced state spaces.
- Can be far more robust for tracking in constrained situations
    1. Unibody: Internally used – solve system as a single rigid body
    2. Duobody: experimental – solve the system as 2 solid parts: arm and hand
    3. Arbitrary joint locking

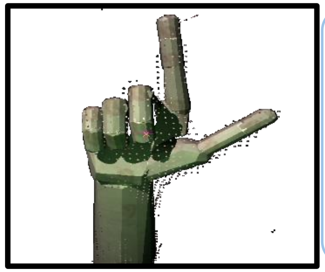
Pose Locking: Best-fit pose given only pointer finger and wrist as open rotational DOF

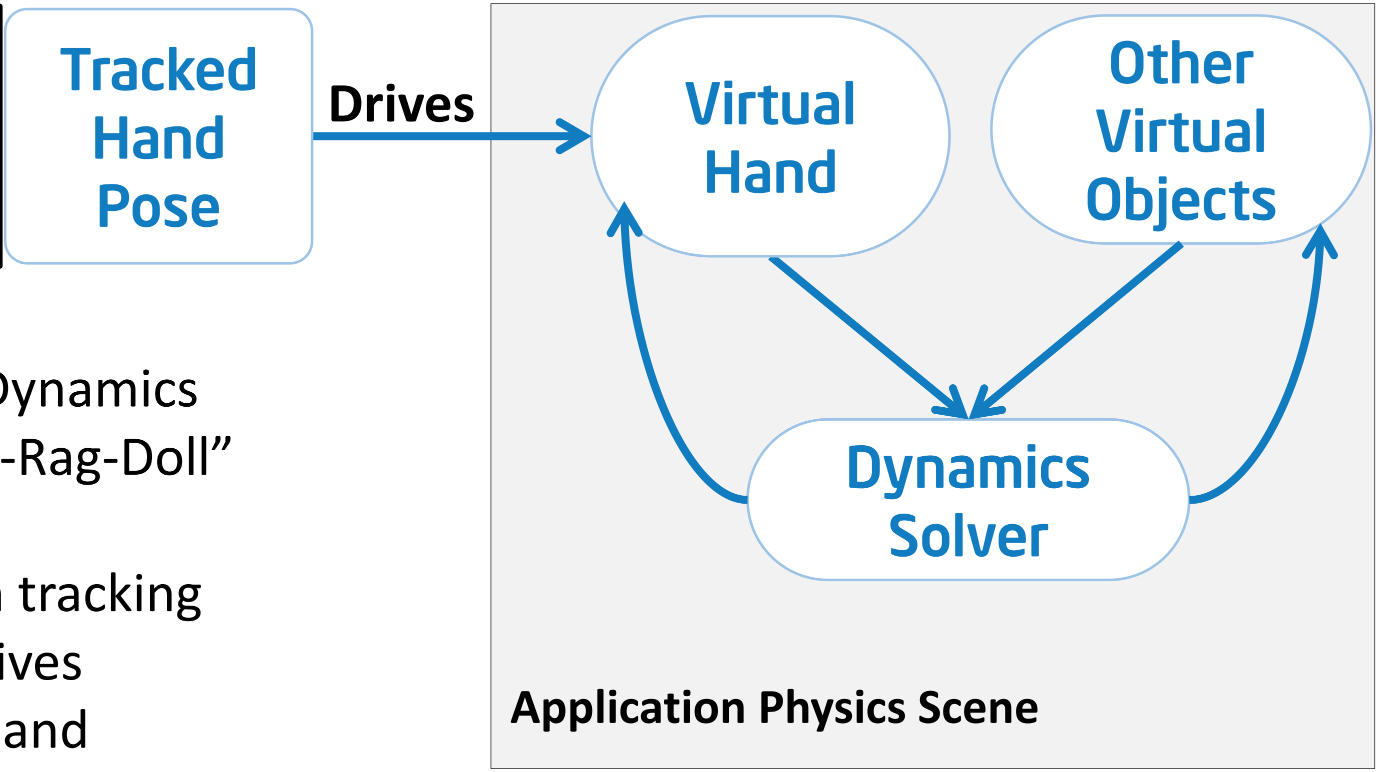


Combining tracking + simulation



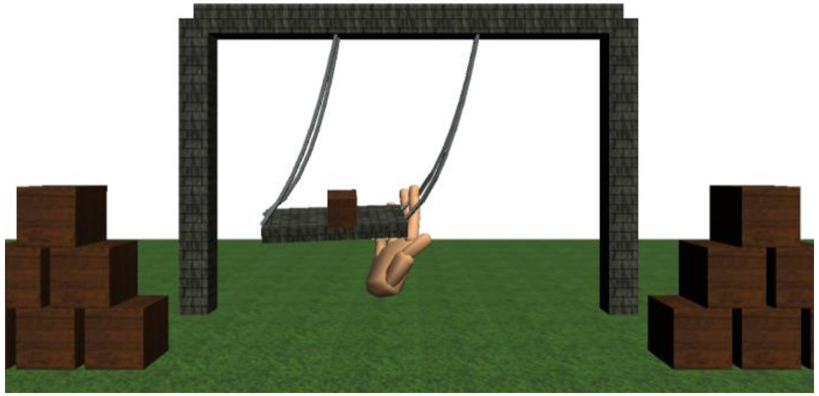
Application – physical 3D interaction



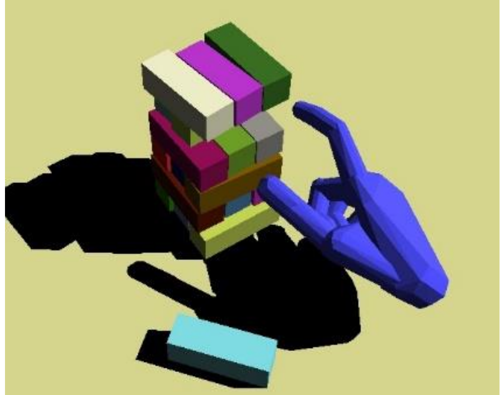


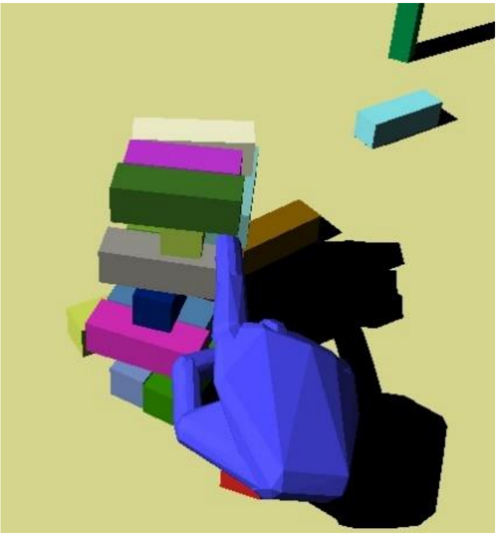
Forward Dynamics
"Powered-Rag-Doll"

Pose from tracking system drives a virtual hand in the application.



Jenga Case Study



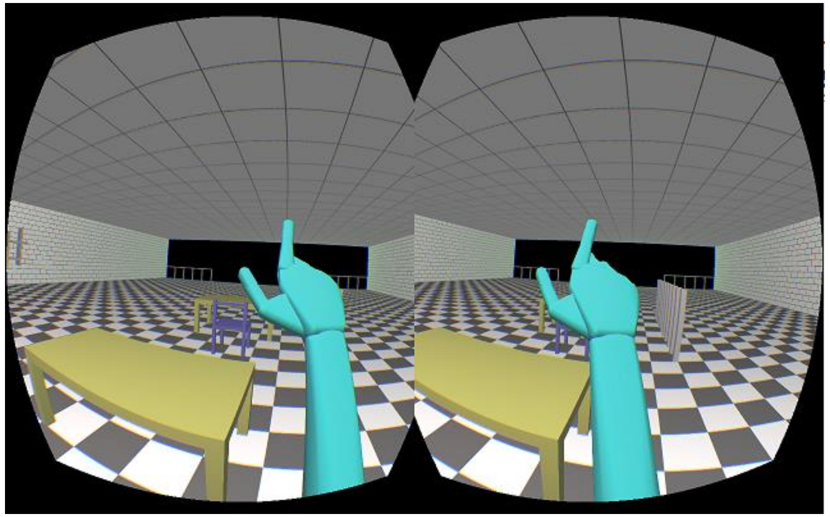


- **Easy** to knock things over, but **Hard** to grasp/stack blocks

- 3D displays helpful for judging distance. (eg zspace)
- Intent-assuming artificial systems can enhance interaction. (extra magnetic pull/push.)
- SoftBody seems to work better than RigidBody (wet bar of soap vs sponge)

- But really too hard to play without force-feedback
  - Interesting area of further work: how to combine tracking systems with force understanding and communication
  - ***Tu-Hoa Pham, Abderrahmane Kheddar, Ammar Qammaz, Antonis A. Argyros***; The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 2810-2819

Applications to HMD applications



GPU vs CPU

- Please build CPU-side tracking

GPU has a ton of high-throughput numerical compute but it has two problems

1. Run to completion on tasks
2. VR+AR applications have high compute needs and you're getting in the way
   - VR is only viable at ~90Hz [Abrash 2014], which is ~ 11ms/frame
   - Users expect visual fidelity, applications will use 8-10ms/frame
   - Budget of 1-3ms/frame if you're using the GPU (e.g. 330 to 1000Hz tracking)
   - If you miss it, you're toast: > 11ms often means 22ms, which is 45Hz visual updates and your users all get sick

Cascaded Hand Pose Regression, CVPR 15, Sun et al. 300Hz on CPU!

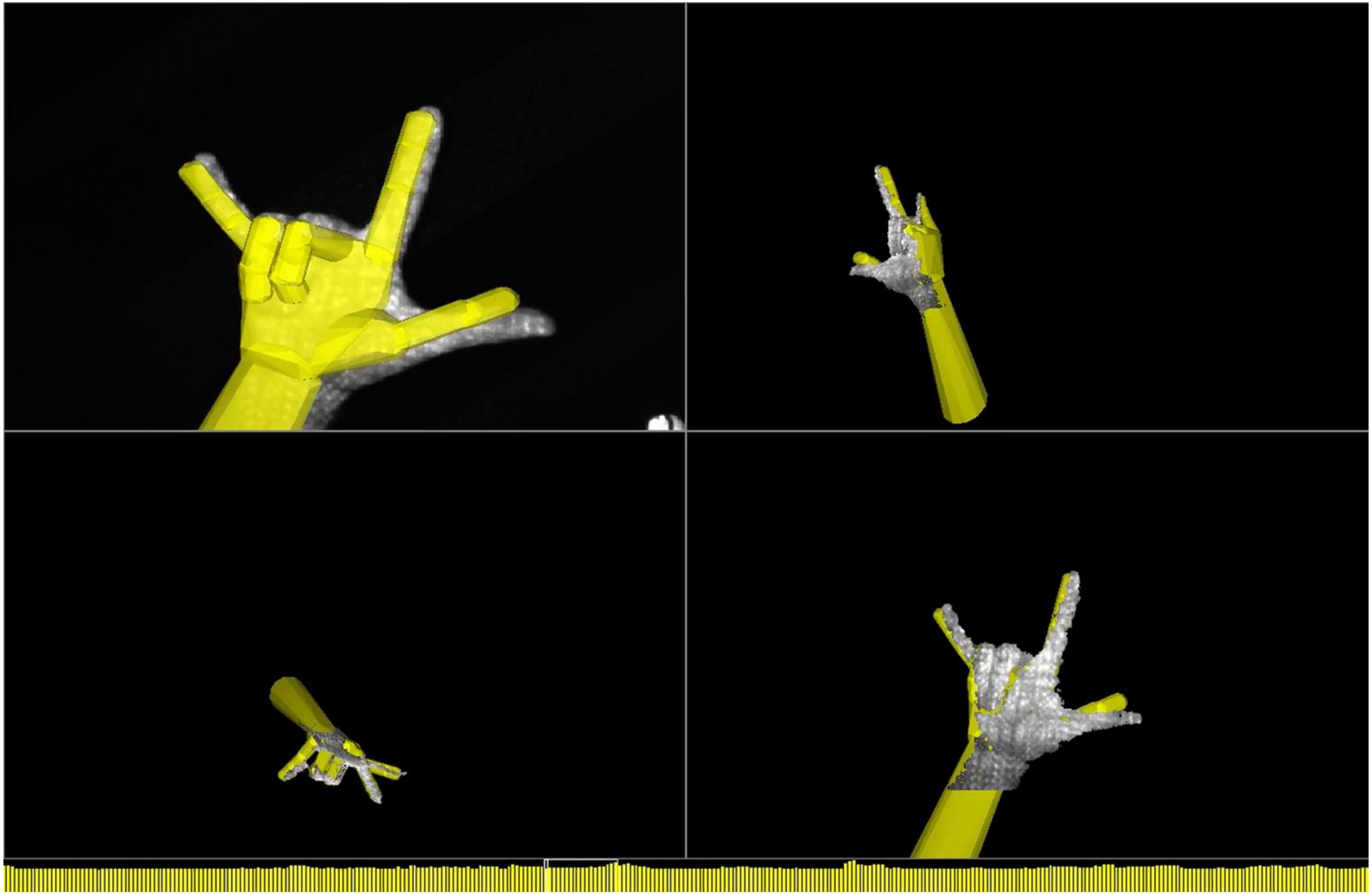
Annotation, Learning & Classification

Getting ground truth

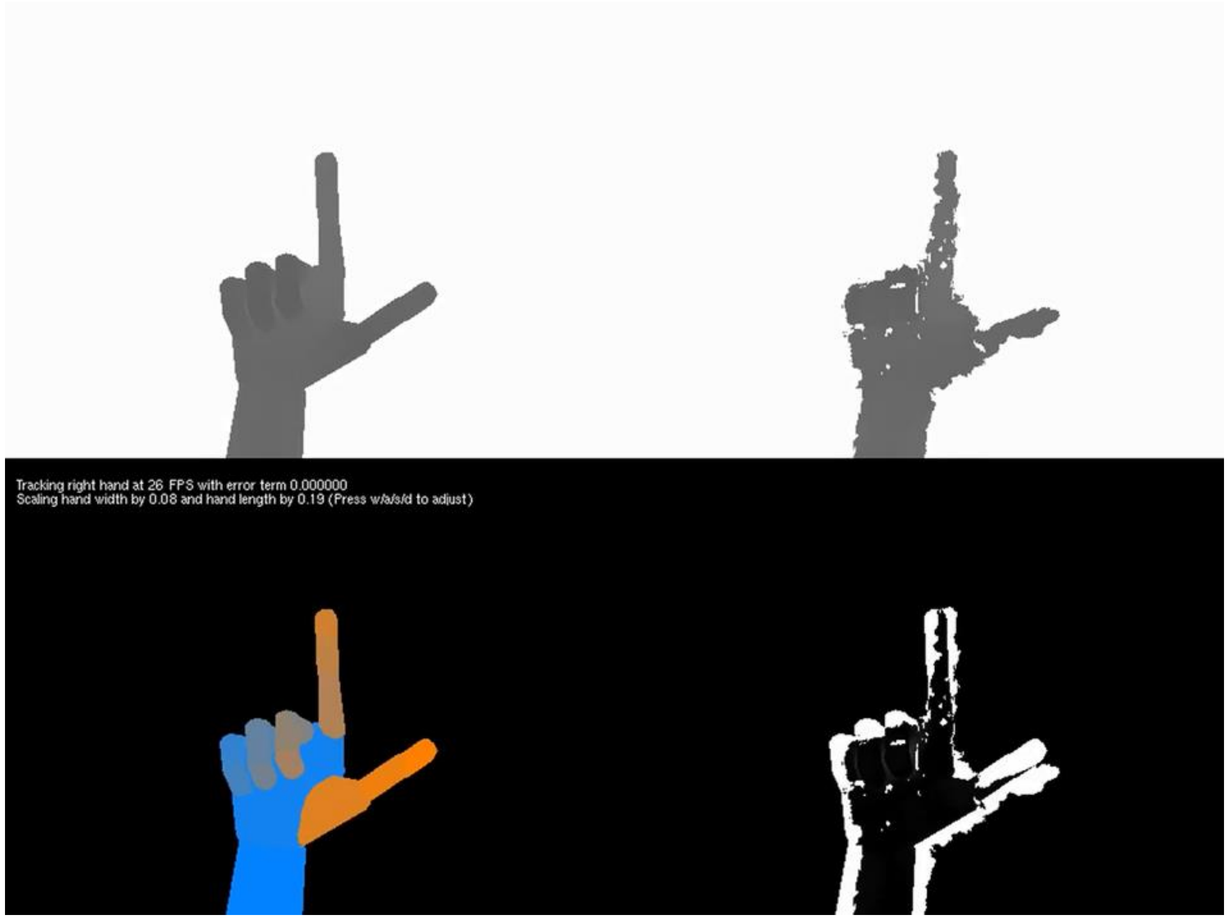
- If you want validation data, or large data-sets for machine learning: use an iterative geometric tracker

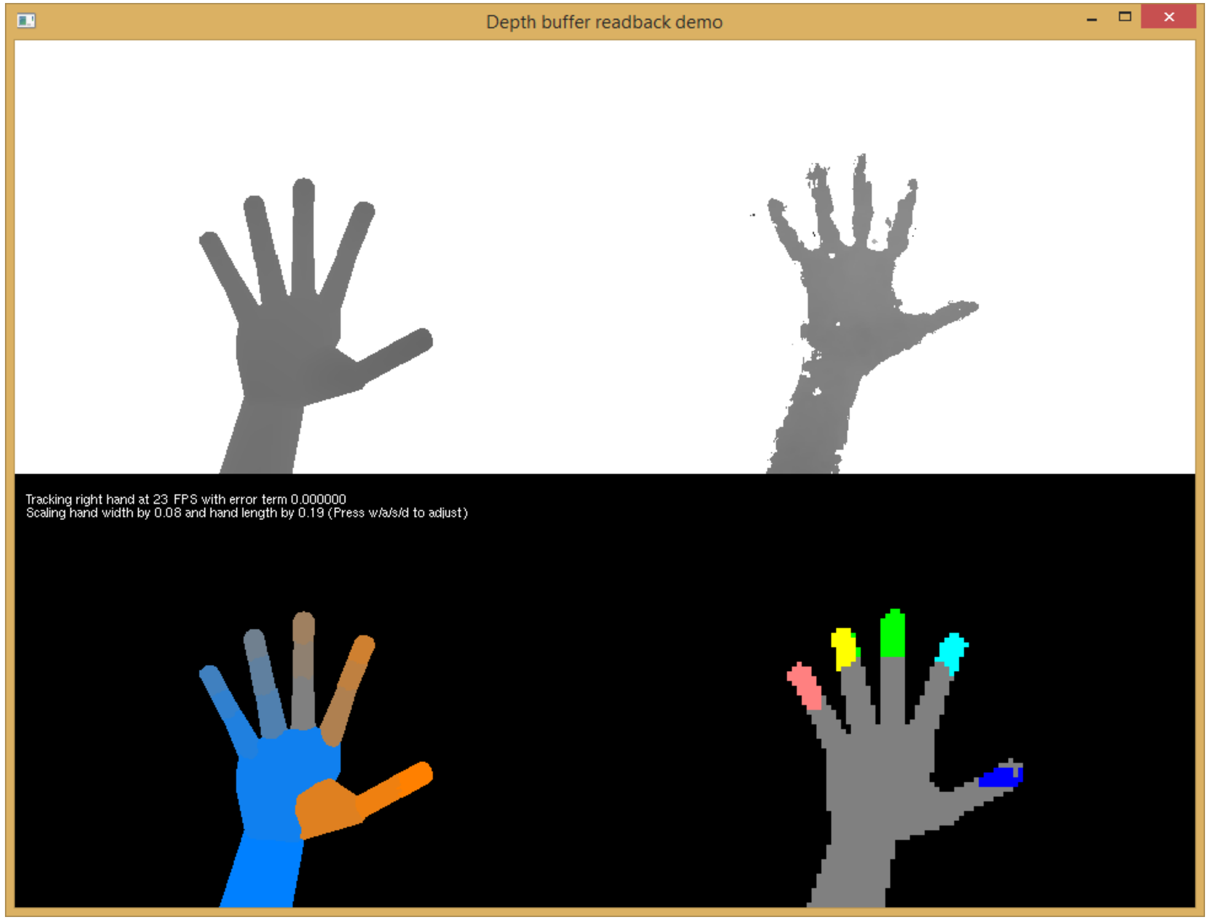
- Failures occur but there's three huge benefits:
  1. Fast and easy
  2. If you annotate a few corrections, you can propagate the corrections
  3. Geometric trackers can handle multiview pose tracking
     - Using multiple cameras, simply register them and feed the algorithm at once
     - We simply minimize cloud -> pose error

- SIGGRAPH 14: "*Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks*" Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin.



Training Label Generation
Left=Annotation, Right=Trained Classifier





Explicit a-prior polyhedral model

Pros

- Not solving unnecessarily high dimensional problem
- Easy to render, collide against

Cons

- Not as high-fidelity as a generic skinned mesh
- Doesn’t handle variation across users

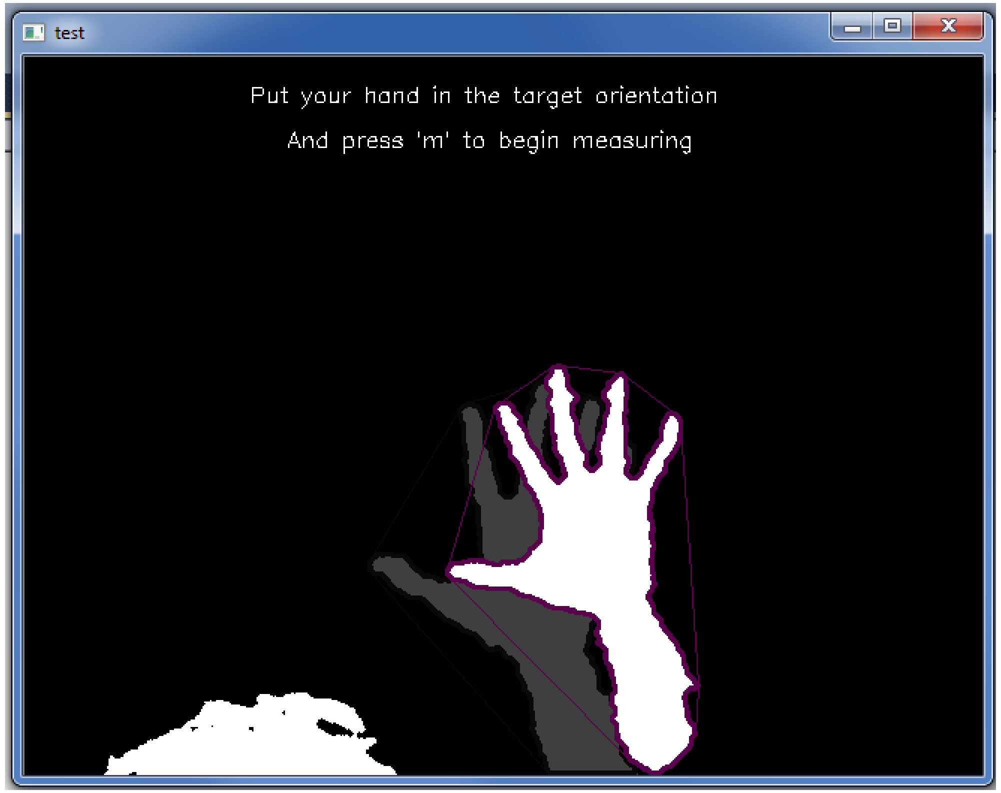
Hand Variation Across Users

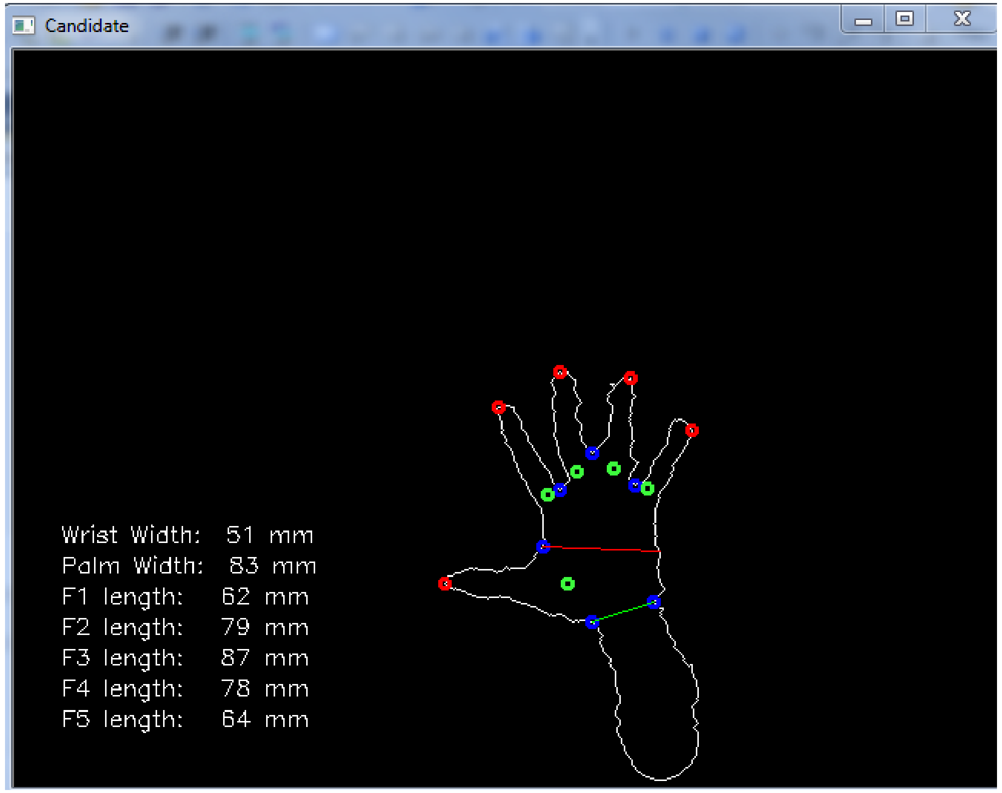
- We’ve found that most adult humans have very similar sized hands
- We’ve been using a simple to use 2-parameter resizing model
  - Length
  - Width/Thickness
  - User-controlled
- CVPR 2015: Sameh Khamis, Jonathan Taylor, Jamie Shotton, Cem Keskin, Shahram Izadi, Andrew Fitzgibbon; “*Learning an Efficient Model of Hand Shape Variation From Depth Images*”
  - captures high level-of-detail variation, and also justifies using just 2 or 3 degree of freedom variation model of hand variation

Hand Variation Across Users

Internal work on naïve hand measurement work done in June 2012

Automatic Hand Measurement: Overview



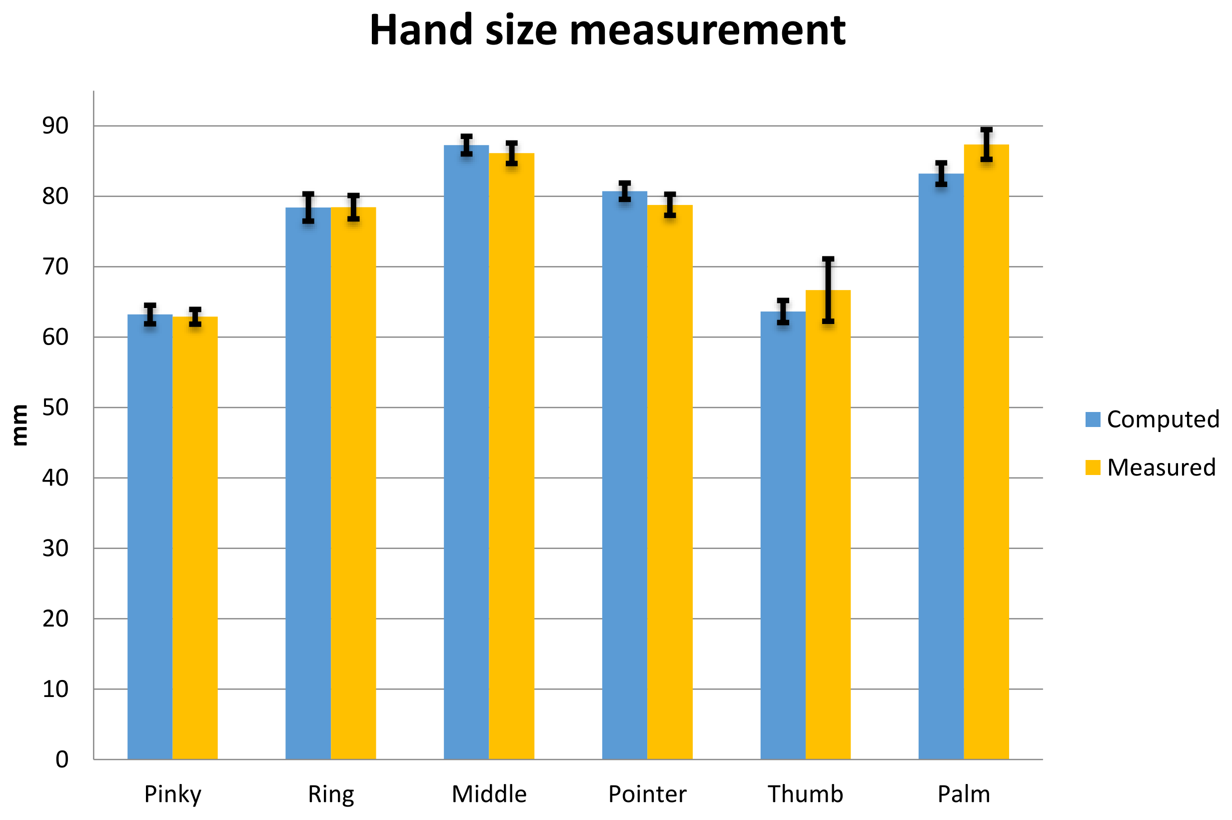


1. Detect blobs, in pre-determined orientation
2. Find points of interest on the contour
3. Feeding a 6 parameter model: finger lengths (5), palm Width

Automatic Hand Measurement: Accuracy

Finger	σ (mm)
Pinky	0.7
Ring	1.0
Middle	0.6
Pointer	0.6
Thumb	0.8
Palm	0.8

	Mean	Median	Mode
Mean |E|	2.3%	2.3%	2.8%

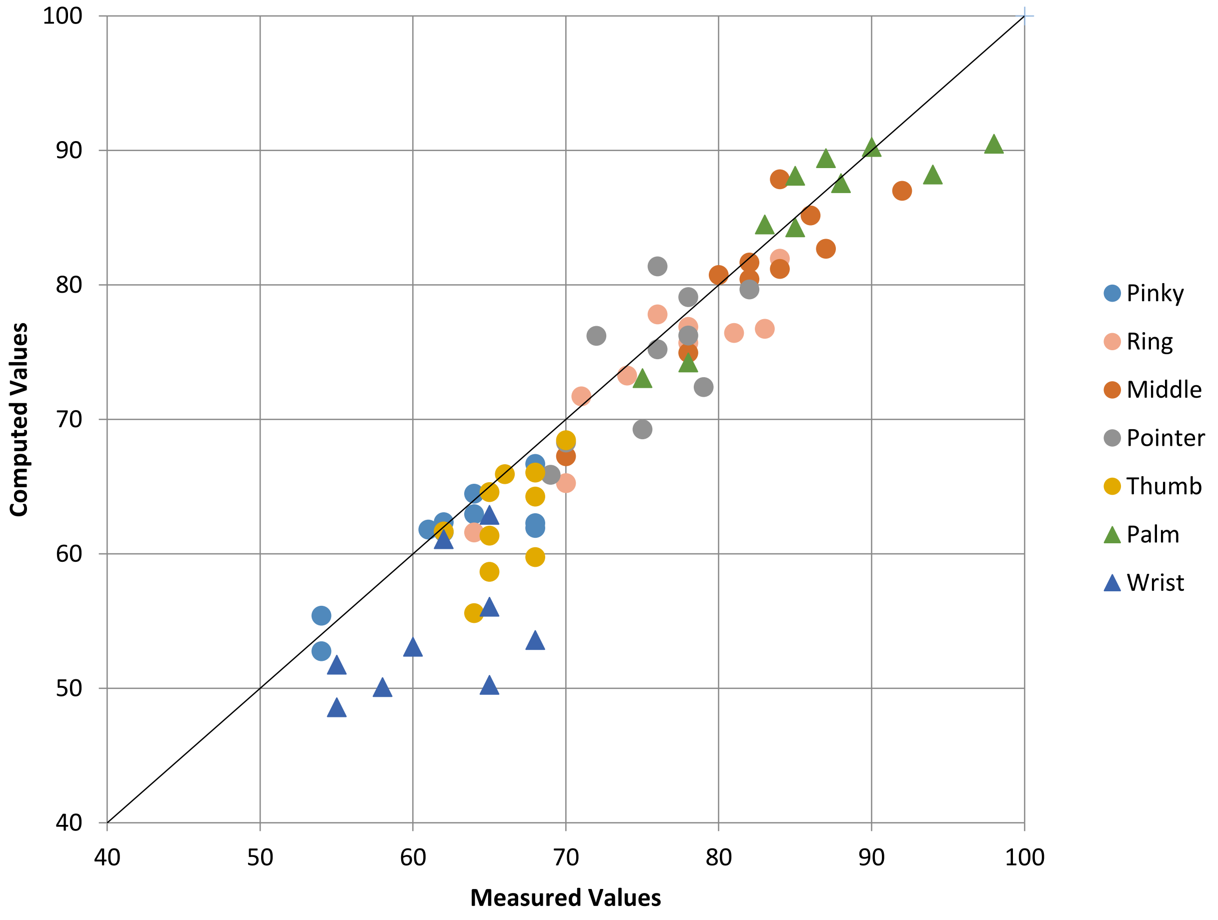


Automatic Hand Measurement: Multiple Subjects

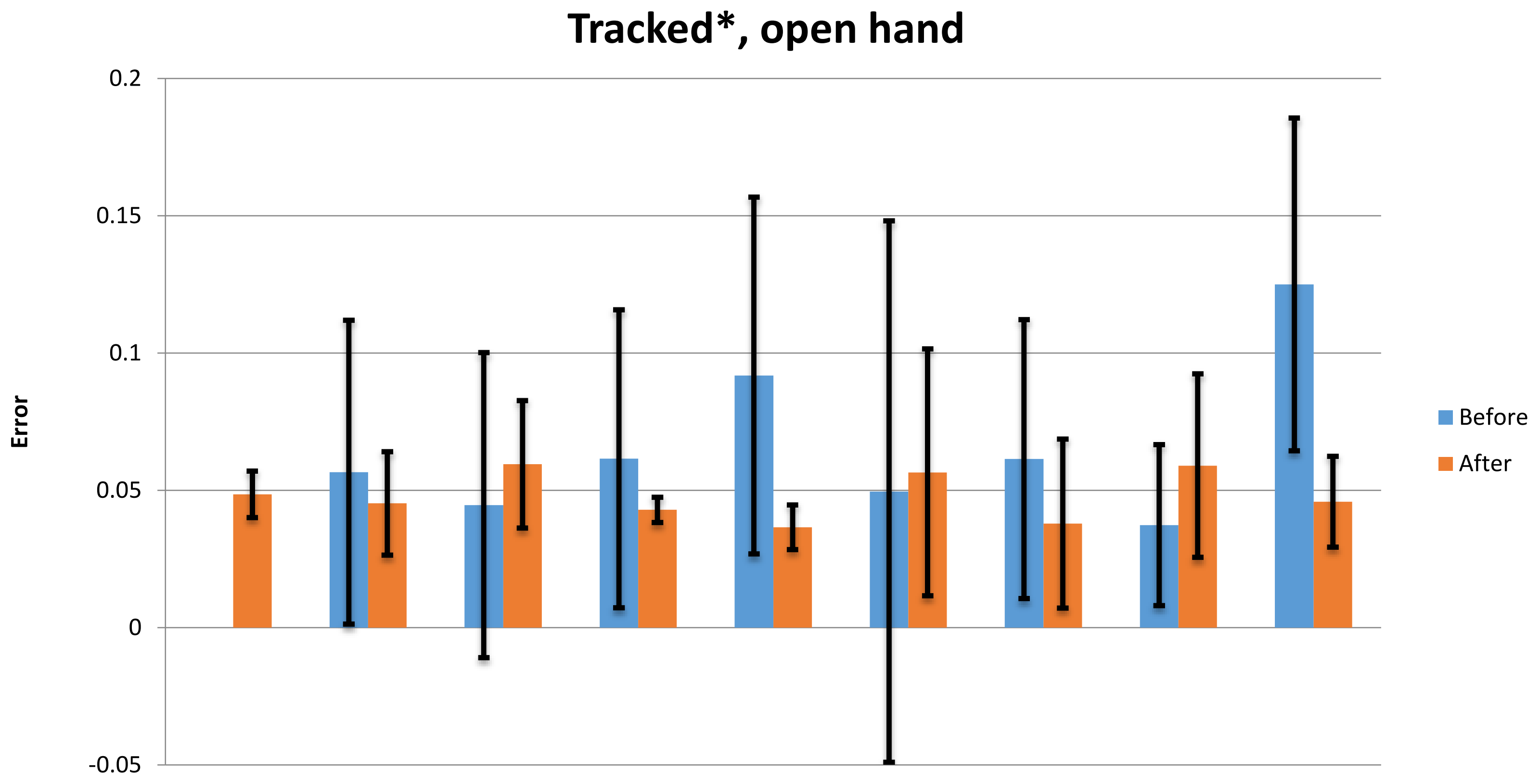
	Measured results (10 users)									
	1	2	3	4	5	6	7	8	9	10
Pinky	68	64	54	62	68	61	64	54	68	62
Ring	84	78	70	76	78	74	83	64	81	71
Middle	92	80	78	84	87	84	86	70	82	82
Pointer	82	76	70	76	79	78	78	69	75	72
Thumb	68	70	65	65	65	62	68	64	66	68
Palm	90	85	75	83	85	94	98	78	87	88

	1	2	3	4	5	6	7	8	9	10	**AVERAGE**
Mean (|E|) *	4.17%	1.19%	4.67%	2.84%	5.65%	2.43%	4.11%	5.40%	4.25%	2.12%	**3.69%**
Correlation	0.971	0.993	0.949	0.987	0.955	0.998	0.965	0.951	0.952	0.973	**0.969**

Automatic Hand Measurement: Multiple Subjects



Automatic Hand Measurement: Applied to hand tracking



	Mean	σ
Before	0.066	0.059
After	0.048	0.021

Q&A: Additional Interactions

Physics Reference Works

Classical Works

- GJK, 1988
- Fast Contact Force Computation, Baraff, 1994
- Anitescu & Porta, 1996
- Impulse-based Dynamics Simulation, Mirtich & Canny, 1994-1996

Modern/Education Works

- Open Dynamics Engine, Russell Smith, 2004
- Iterative Dynamics with Temporal Coherence, Erin Catto, 2005
- Modeling & Solving Constraints, Erin Catto, GDC 09
- Physics for game programmers, *GDC 2012*
- Understanding Constraints, Erin Catto, GDC 2014
- Exploring MLCP solvers, Erwin Coumans, GDC 2014