# Robotic Grasping: A Data Driven Approach

Peter Allen
Columbia University

Collaborators:

Matei Ciocarlie
Corey Goldfeder
Hao Dang

# Data-Driven Computation

- Data-driven approaches solve hard problems
- Simple machine learning algorithms often outperform more sophisticated ones if trained on large enough databases.
- Natural language translation, semantic annotation, image reconstruction
- Bigger databases are as important as smarter algorithms
- Computer Vision example: Have we seen everything?

Image completion, Haye and Efros, Siggraph 2007

# Is Grasping Indexable?

- Many previous attempts to taxonomize grasps
- Is there a finite set of grasps we can pre-compute?
- If so, can we build an indexable database of grasps?
- Given a novel object to grasp, can we find a similar grasp?
- Some Problems:
  - Lots of objects to grasp…
  - Lots of DOF in a hand (~20 + 6 in human hand)…
  - Lots of robotic hands…
- Intractable?  But maybe not….

# Key Ideas: Data-Driven Grasping

- How do we generate large amounts of grasping data?
- Turn robot on in lab, feed it objects to be grasped 24/7?
  - Impractical, and hardware is not that robust
- Get lots of humans to grasp lots of objects and record data?
  - Still limited # of subjects and # of objects
- Solution: Generate lots of data through simulation
- Use low-dimensional subspaces to make problem tractable
- Use a very large corpus of 3D objects to be grasped

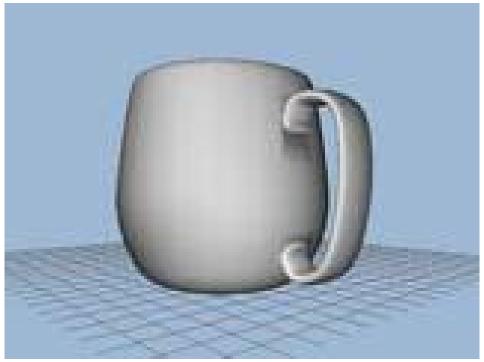
# Grasp Planning in Simulation

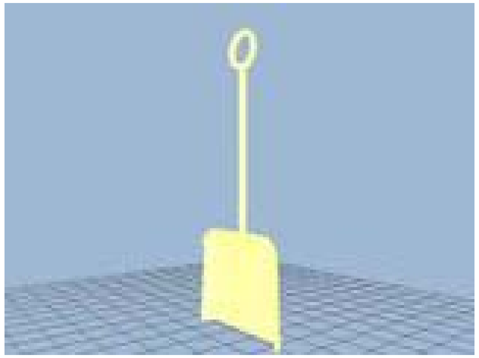
- Simulation is fast and cheap
  - And allows numerical quality measures
- Offline planner is the Eigengrasp Planner*
  - Reduced dimensionality control space derived from human trials
  - Simulated annealing on a grasp energy function
  - Evaluate candidates in GraspIt! Simulator
  - Converges on a good grasp

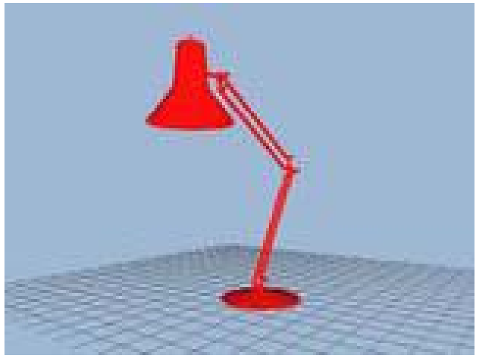
*Ciocarlie, Goldfeder, Allen IROS 2007

# Finding Objects to Grasp

- We use the 3D models from the Princeton Shape Benchmark*
  - Well known academic dataset of 1,814 models
  - All models resized to "graspable" sizes
- We provide grasps at 4 scales
  - …because grasping is scale dependent
  - 7,256 3D models in all

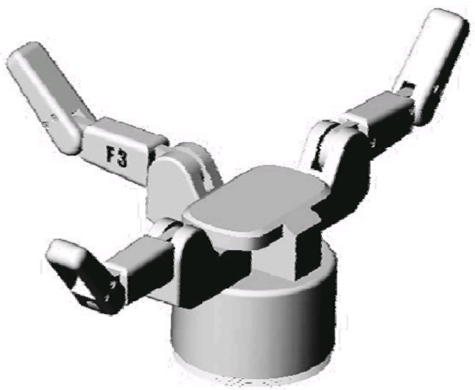


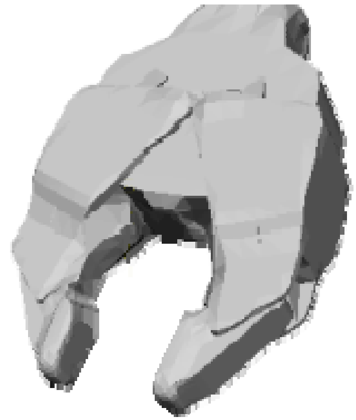




# Robotic Hands

- We provide grasps for 4 hands
  - Human hand model (20 DOF)
  - Barrett Hand (4 DOF + disengaging clutch)
  - Barrett Hand with rubber coating
  - Willow Garage gripper
  - More hands to come!

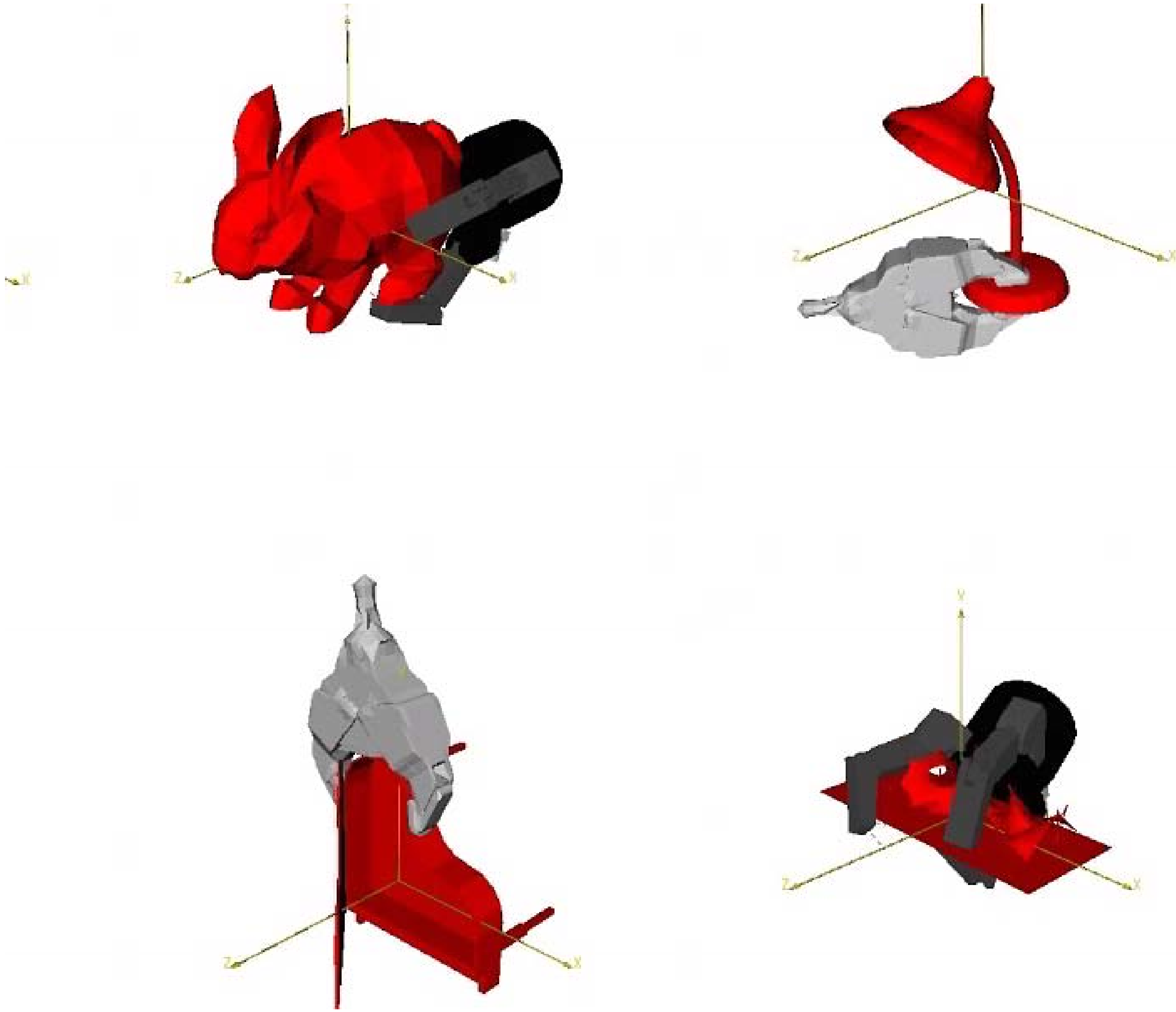


# Building the Columbia Grasp Database

- Simulated annealing in a eigen-grasp space
- 8 dimensions: 6 pose + 2 eigenvectors
- 1,814 objects at 4 scales =7,256 objects to grasp
- Grasps evaluated in *GraspIt!* simulator for 4 hands
- 6 compute-months on multicore workstations
- Contains over 250,000 form-closure grasps
- Includes pre-grasp poses, contact points, and Ferrari-Canny quality metrics
- A new tool for the grasping community
- Available at grasping.cs.columbia.edu

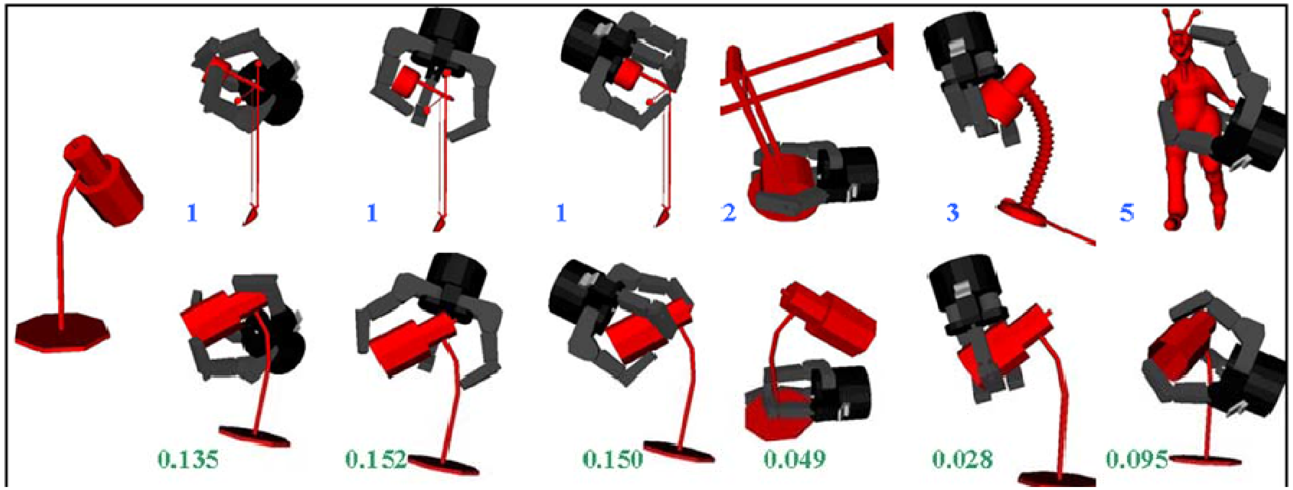
# Browsing the CGDB*



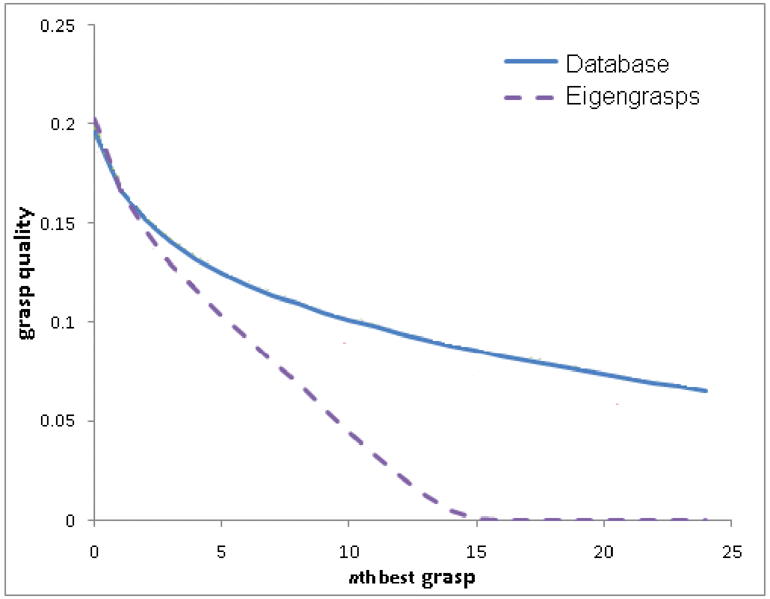
*Goldfeder et. al., ICRA 2009

# Data Driven Grasp Planning

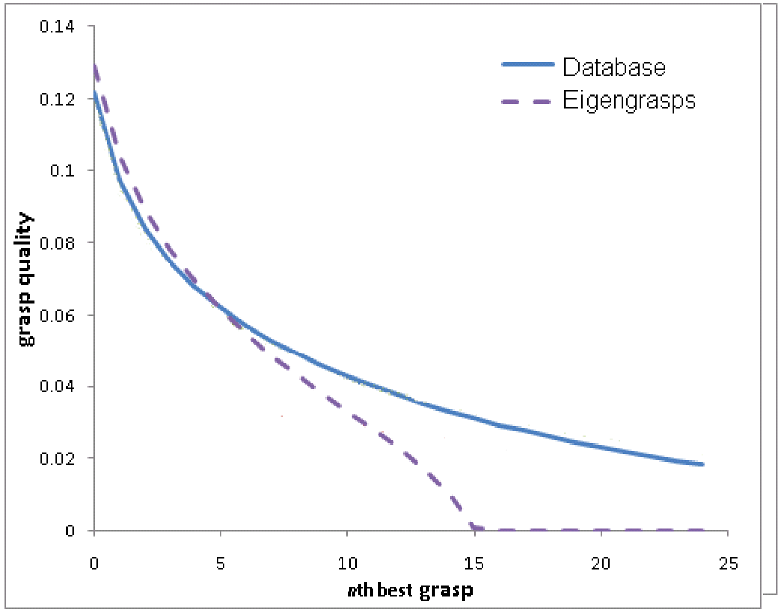
- Given a new 3D model to grasp
  - Find nearest geometric neighbors in database
  - Can use choice of shape matchers
  - Collect pre-grasps from neighbor models
  - Evaluate candidates in GraspIt! simulator



# Benchmarking Database Planner



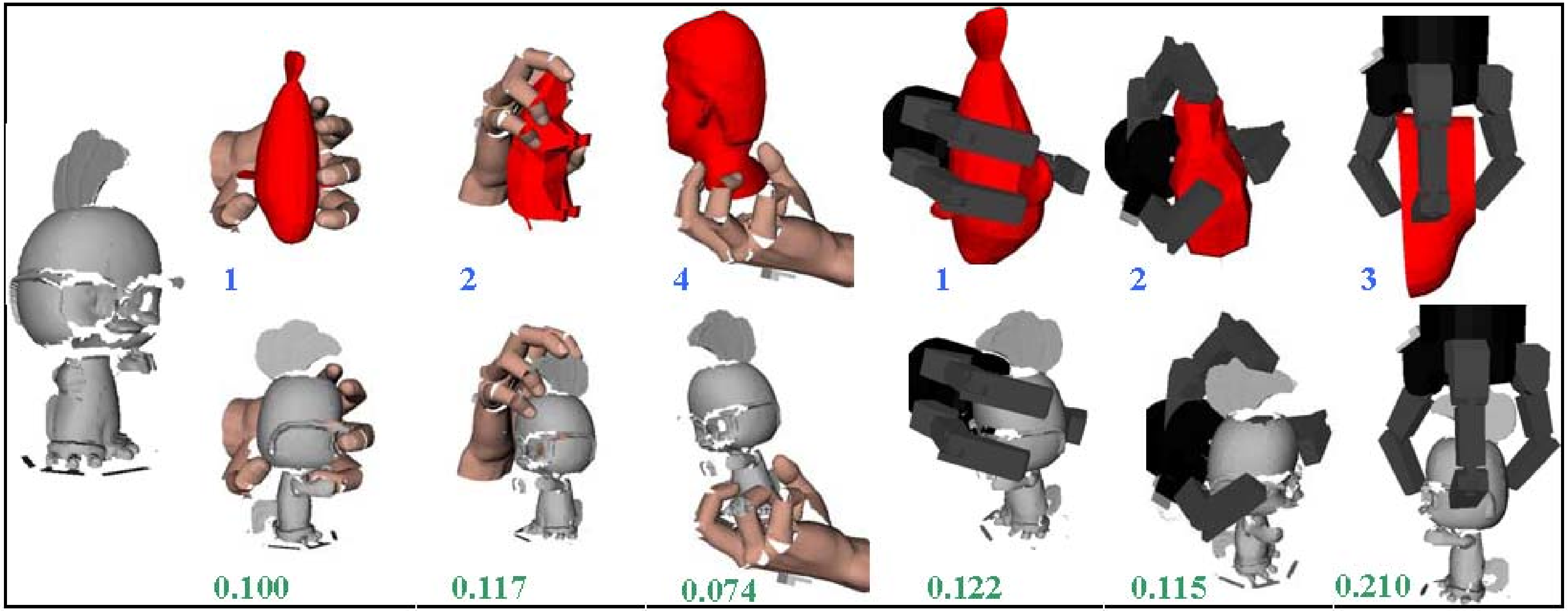

Barrett Hand




**Human Hand**

- $n^{th}$ best grasp for data-driven & Eigengrasp planners
- Database planner is also 30x faster

# Grasping Novel Objects: Noisy Range Data




- Caveat: Geometric match needs full 360° scan model - not usually available
- Can we grasp with only partial sensor data?

# Matching with Partial Sensor Data

- Traditional approach: register point clouds into a 3D model
- What kind of range sensor data can we really expect?
  - Noisy, inaccurate depth measurements
  - Imperfect registration between scans
  - Sensitive to noise and occlusions
- Better idea: match directly in sensor image space
- Find view dependent features for matching

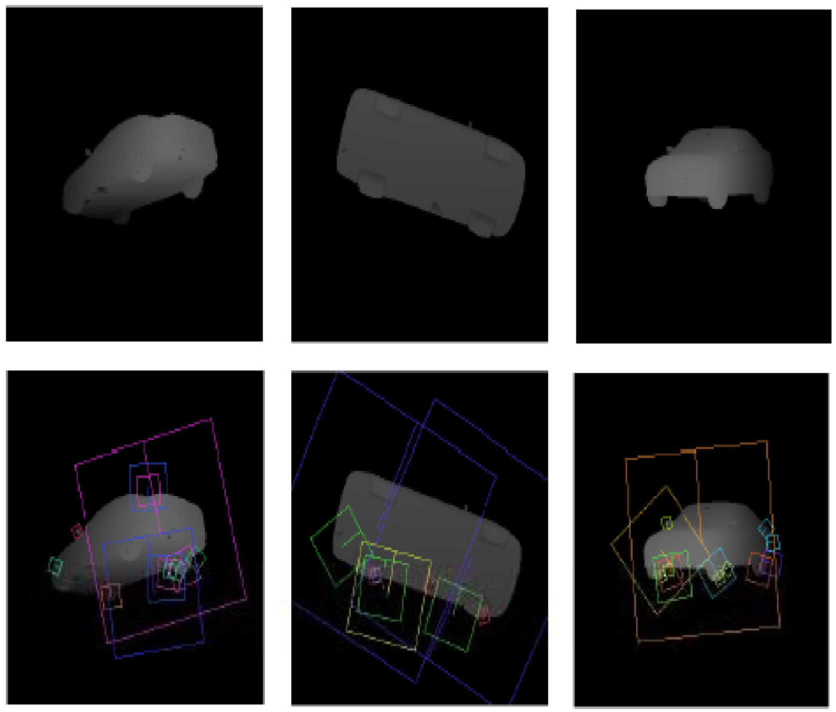
# Data Driven Grasping Pipeline

- Acquire partial-view depth data
  - Must be realistic – sensors can’t see full models
- Match into database
  - Using only the acquired partial data
- Align the sensor data to the model
  - Using only the acquired partial data
- Rank grasps from matching models
  - How do we know which candidate to output?

# Depth SIFT

- 3D matching with Depth SIFT*
- Compute SIFT on depth images from many views
- Describe all views with single "bag-of-features" histogram
- Only sees depth *gradients*, more stable than depth
- Operates in natural space of sensed depth data
- Registration-free matching of multiple scans

Honda Accord
(found on web)



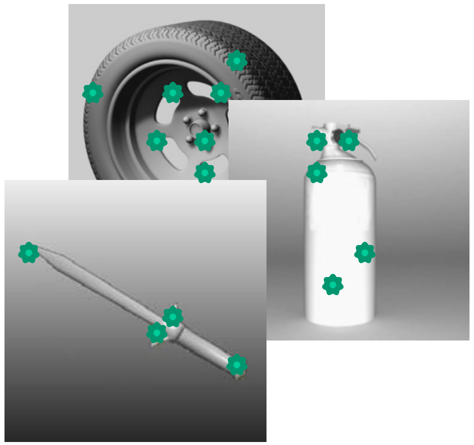
Depth
Images

Sift
Features

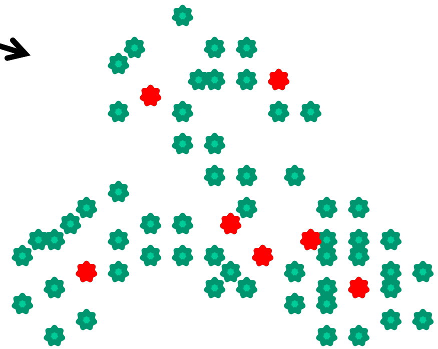
* Ohbuchi *et al.*, SMI 2008

# Training Depth SIFT

For *n* training models



Render depth images and compute features



Gather SIFT features from all images

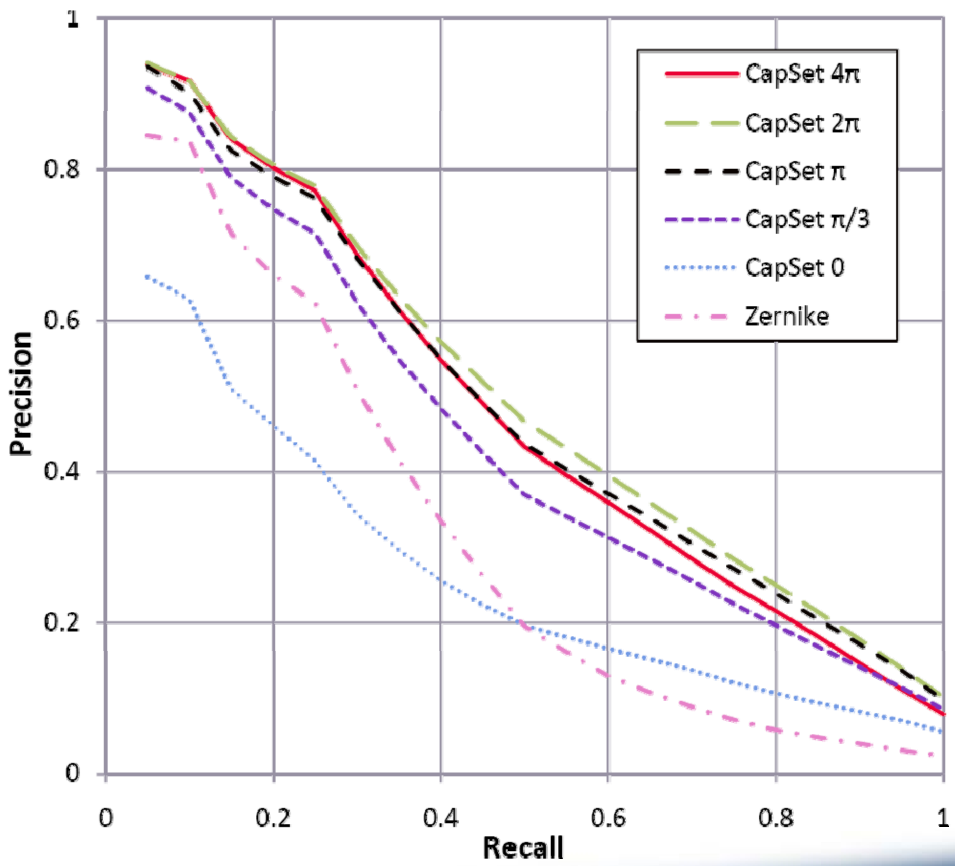
Cluster into "codebook"

# Matching Using the Codebook: *Caps* and *CapSets*

- Extension of Depth SIFT to partial data
- Combine views from a spherical cap rather than a sphere
- Create binary vector of codebook features
- Models what can be sensed realistically
- $Cap_\theta(v)$ is a function of solid angle θ and center view $v$
- A $CapSet_\theta$ collects the $Cap_\theta(v)$ descriptors of an object from all $v$
- $CapSet_{4\pi}$ is identical to Depth SIFT
- Compare models by histogram similarity
  - Similar models should have similar histograms
  - We use binary histograms and Jaccard distance

# Precision/Recall of CapSets

- *CapSet*$_\pi$ is indistinguishable from Depth SIFT
  - And only needs to see ¼ of a model to match!
  - Almost as good with smaller view clusters
- Bottom Line: Just need to take a few scans around a viewpoint to get good performance
- Actively move your sensor!



# Aligning Partial Scans

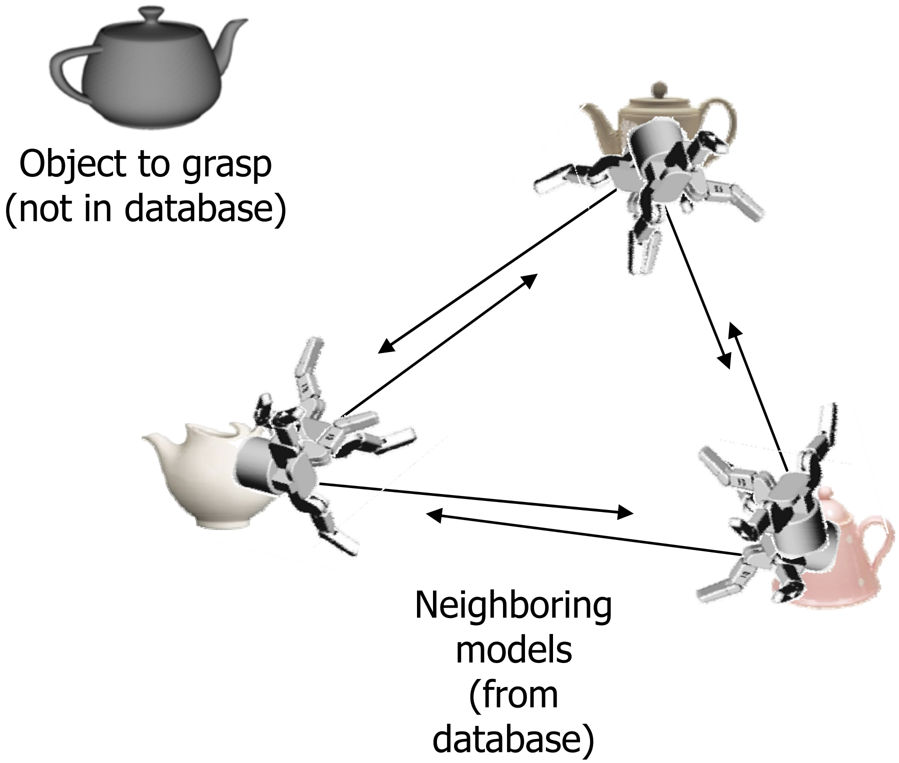
- To transfer grasps, we need alignment
  - Align full 3D models (from the CGDB) with partial sensor data
  - PCA (used in our previous work) isn't applicable to partial data
- *Cap*-based alignment method
  - Find the *Cap* of the CGDB model that best matches the sensed data
  - Align the center view of that *Cap* with the most central view from the sensor
  - Use geometric moments to resolve the roll angle
  - Refine the alignment with ICP

# Grasp Selection

- Once we match geometrically, we can retrieve many candidate grasps from nearest neighbors
  - But we can’t simulate and test their quality on partial data!
  - How do we decide which grasp to use?
- Suppose we have a candidate grasp for a model
  - How *generalizable* is it to similar models?
  - We can try it on similar models in the database and see!
- We rank grasps by cross testing them
  - We can do this offline (less accurate but faster planning) or online

# Online Cross Testing

- At grasp time:
  - Find the object's *k* neighbors
  - Try grasps from each neighbor on the neighbors of the object
  - Measures how well the grasp generalizes in the neighborhood of the sensed object

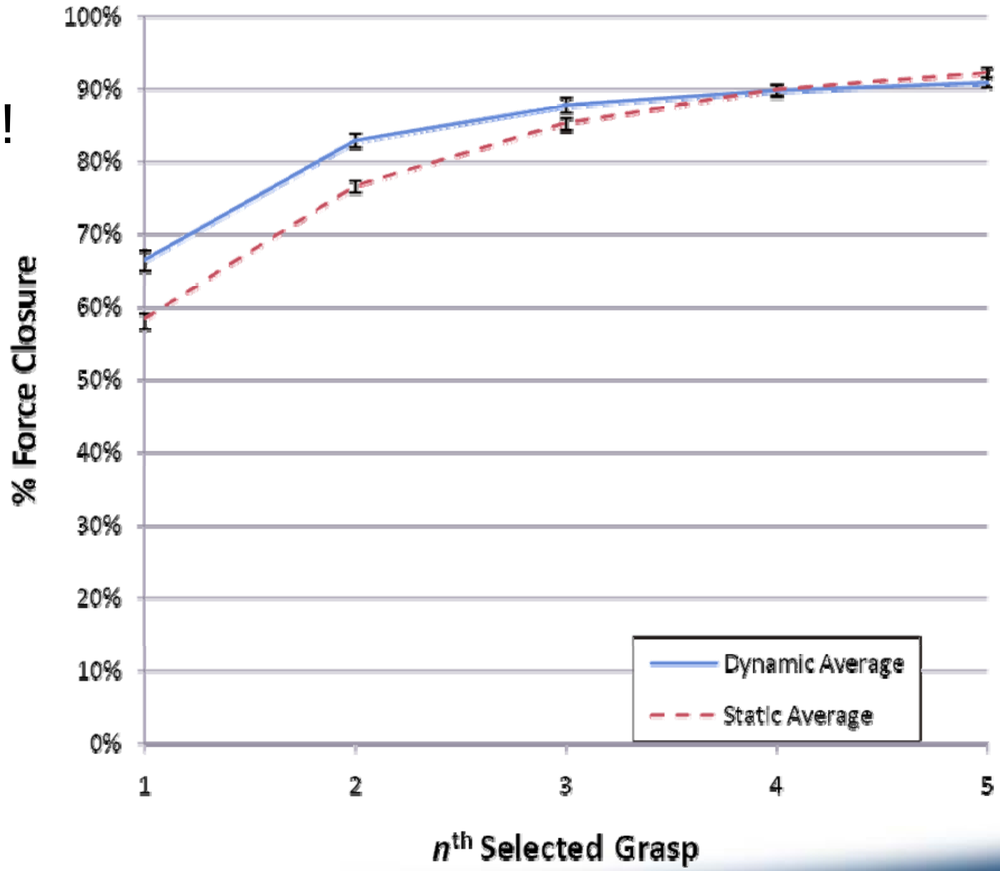


# Full Pipeline

- Acquire a set of clustered range scans
  - Feasible for real robots in real time
  - No need for 360° scan coverage
- Match into database using *Cap* descriptors
  - No need to register scans!
  - The more views, the better you can match
  - But degrades gracefully to even a single view
- Rank grasps from matching models
  - Using the generalizability criterion
- Align the grasp to the partial data
  - Using our new alignment method

# Results: Simulation

- Experiments with a Barrett hand and *CapSet*$_{\pi}$,
  - Ranking with online cross testing
  - Results averaged over 1,814 models in the CGDB
  - Very difficult models to grasp!
  - Grasp quality evaluated statically (conservative) and dynamically (more realistic)



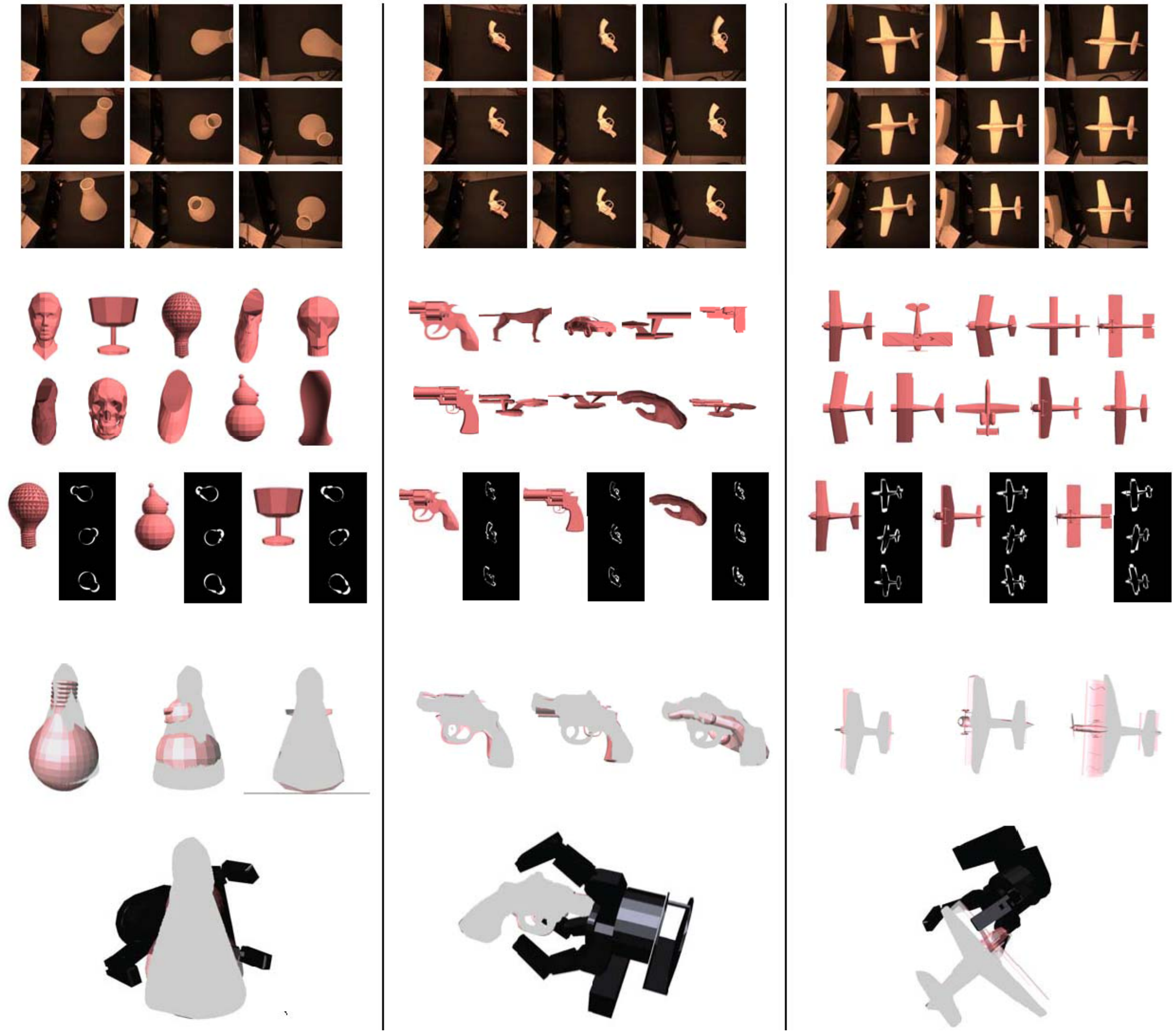

- 6 experiments per model
  - 6 different viewpoints
  - Note the tiny error bars
  - 83% form closure within first 2 grasps
  - 77% in static analysis

# Results: Mobile Manipulator

- HERB mobile manipulator, Intel Pitt.
- Uses single camera, not rangefinder
- Depth SIFT not feasible
- Added visual hull silhouette shape context features instead of SIFT features to Capsets
- Scenario:
  - Take 3 images of object over $\Pi$ steradians
  - *Cap*$_{\Pi}$ used to find 10 closest models
  - Each model aligned with data, reordered by best fit
  - Use cross-testing to find best grasp
  - Transfer grasp to object

# Partial View Grasping

1. Take images of unknown object
2. Partial view shape match
3. Align with Silhouettes
4. Choose best aligned model
5. Index grasp in CGDB



# Experiments

# Conclusion

- Promising new approach to grasping
- Modular:  add your objects, your robotic hand, your sensor features, your shape matcher
- Problems:
  - calibration important
  - material properties, mass assumed

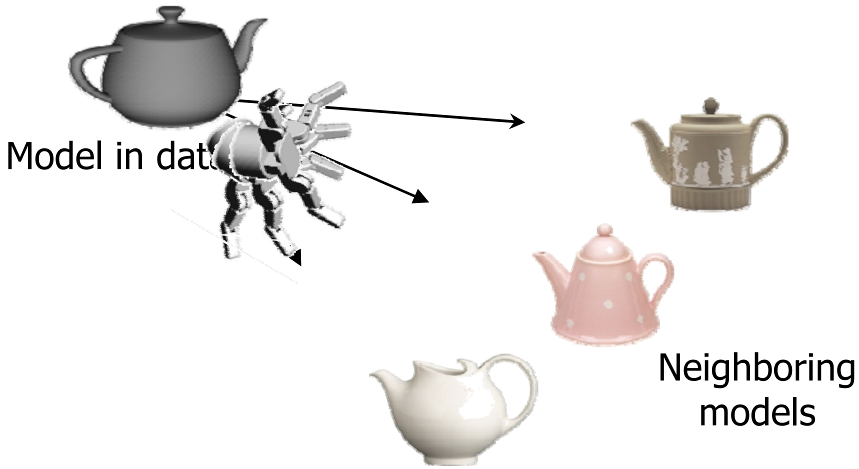
Acknowledgements:

- NIH  grant 1RO1 NS 050256-01A2
- NSF Grant 0904514
- Sidd Srinavasa, Intel Labs Pittsburgh

Examples:
Grasping Novel Objects

# Offline Cross Testing

- During training:
  - For each *database* model
  - Score the model's grasps by how well they transfer to the model's own neighbors




- At grasp time:
  - Find the object's neighbors
  - Rank grasps from all neighbors by precomputed score

# Full Pipeline Example

Depth images of a real wineglass, acquired with a NextEngine scanner

Models from the CGDB with similar $Cap_{\Pi}$ descriptors

Alignments with the partial model of the wineglass

The first 5 grasps from cross-test ranking

# Using Depth SIFT

- For each model:
  - Render depth images from sample views
  - Compute SIFT features of these images
  - Assign each feature to the best "representative"
  - Output a histogram of which representatives appear

- Compare models by histogram similarity
  - Similar models should have similar histograms
  - We use binary histograms and Jaccard distance