ROB 498/599 · 3D Robot Perception · Fall 2026

# L12 Object-centric perception

*One object at a time: its 6D pose by render and compare, the side the camera never saw, and the parts that move.*

Wed Oct 14, 2026 • WRITE-IN build

# Object-centric perception

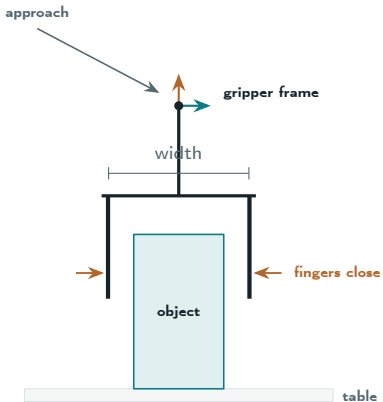
*L12 · Unit 2*

*L11 labelled every point in the map. Today: one object. Where is it, what is on the side we never saw, and how do its parts move?*

# Today

1. What a robot grasp is, and why it needs the object's 6D pose.
2. The classical answer: matched points, point pair features, ICP.
3. FoundationPose, the response paper: one network for unseen objects.
4. Scoring a pose, including symmetric objects.
5. Hidden shape, moving parts, deformables, affordances.

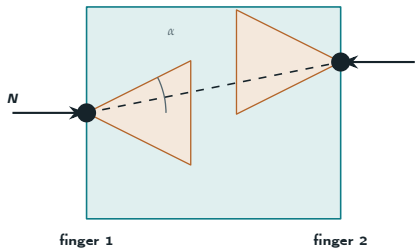
# What is a grasp?




- The most common robot hand: a **parallel-jaw gripper**, two fingers that close toward each other.
- A grasp is a **gripper pose**: where the gripper frame sits (3 numbers) and how it is turned (3 numbers), plus the finger opening.
- Execute it: move the open gripper to that pose along a clear path, close the fingers, lift.

**How many numbers does a robot have to choose for one grasp?**

# What makes a grasp hold

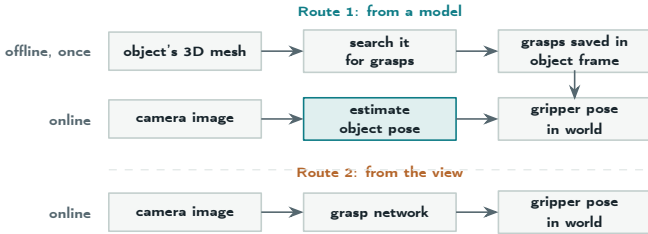



A finger pushes along the surface normal with force $N$. Friction adds up to $\mu N$ sideways ($\mu$: friction coefficient). So the finger's force can point anywhere in a **friction cone**, half-angle $\alpha = \arctan \mu$.

**Antipodal grasp:** the line between the two contacts lies inside both cones. Squeezing then pins the object.

■ $\mu = 0.5$. **The contact line is 11.3° off the normal. Does it hold? At 35°?**

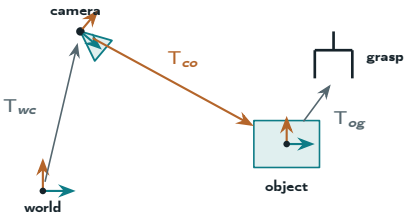
# Two ways to get a grasp




- **Route 1:** offline, find grasps on the object's mesh and save them in its frame. Online, find the object's **pose**: it carries those grasps into the scene.
- **Route 2:** a network reads gripper poses off a depth image. No pose needed, but it cannot tell which part is the handle.

**The rest of today: Route 1's highlighted step, the object's pose.**

# A grasp is stored in the object's frame




- $T_{og}$: the grasp, chosen once on the object model.
- $T_{wc}$: the camera in the world, from calibration.
- $T_{co}$: the object in the camera. Perception's job: six numbers.

**Where does the gripper go, in the world frame?**

# Three settings, by what you are given

| Setting | Given | Example |
|---|---|---|
| Instance-level | the exact objects, at training time | train on 21 YCB objects, test on the same 21 |
| Category-level | objects of a known category, never this one | any mug, after training on many mugs |
| Novel, model-based | a CAD model, at test time only | a new part arrives with its mesh |
| Novel, model-free | a few photos of it, at test time only | the robot walks round a new object once |

*Each row needs less. FoundationPose handles the last two.*

# The novel-object problem

*inputs and output*

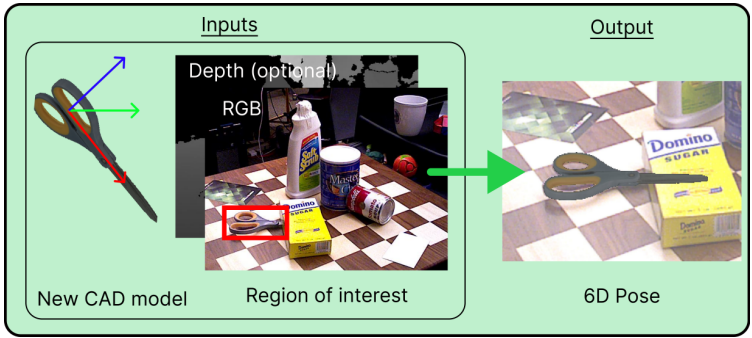



Figure: MegaPose (Labbé et al., CoRL 2022), Apache-2.0

In: an unseen object's model, an image, a box. Out: a 6D pose.

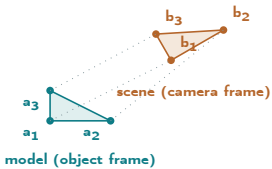
# The same points in two frames




- **Model points** $a_i$: on the object's mesh, in the **object's frame**. Known in advance.
- **Scene points** $b_i$: the same points measured by depth (L02), in the **camera's frame**.
- Same point, two frames: $b_i = R\,a_i + t$. That $R, t$ *is* $T_{co}$, the pose.
- No world frame needed. The camera's place in the world, $T_{wc}$, comes from calibration.

# Procrustes: pose from matched points

**Procrustes alignment:** given matched pairs $(a_i, b_i)$, find the rotation $R(\theta)$ and translation $t$ that line them up best: $\min \sum_i \|b_i - R(\theta)\, a_i - t\|^2$.

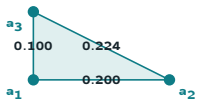



Subtract each set's centroid and $t$ drops out: solve $\theta$, then $t$. Closed form, no iterations.

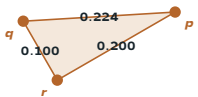
**In 2D (a top-down view), metres. Find $\theta$ and $t$.**

# Which point is which?

Procrustes needs pairs: which scene point b is which model point a? Depth gives thousands of unlabelled points.



model (labelled)



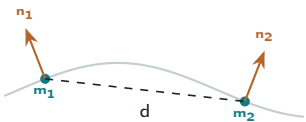
scene (unlabelled)

One thing survives the move from object frame to camera frame: the **distance between two points**. Rotating and shifting an object does not stretch it.

**Using the side lengths, which of $p, q, r$ is $a_1$, $a_2$, $a_3$?**

# Point pair features: matching by lookup table

*Drost et al., CVPR 2010*



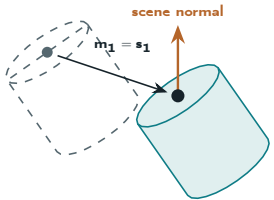

Describe a pair of surface points $m_1, m_2$ by **four numbers**: the distance $\|d\|$, and the three angles among $n_1, n_2, d$ (n: which way the surface faces). Like distance on the last slide, moving the object changes none of them.

**Offline, once per object:** compute the four numbers for every pair of mesh points, and file each pair under them:

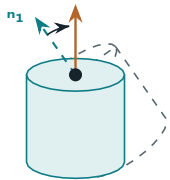
| $\|d\|$, three angles | mesh pairs filed there |
|---|---|
| 5 cm, 90°, 90°, 0° | $(m_{17}, m_{40})$, $(m_{23}, m_{88})$ |
| 8 cm, 60°, 110°, 70° | $(m_5, m_{61})$ |
| ... | ... |

**Online, each image:** a scene pair measures 5 cm, 90°, 90°, 0°. Look it up: it can only be $(m_{17}, m_{40})$ or $(m_{23}, m_{88})$. Two candidate matches.

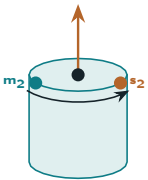
# Why one matched pair gives a pose




1. put $m_1$ on $s_1$: 3 numbers
it can still tilt any way




2. line up the normals: 2 more
it can still spin, like a top



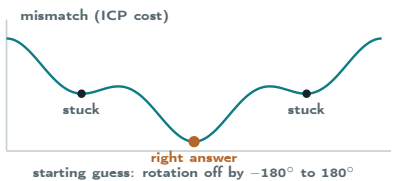

3. spin $m_2$ onto $s_2$: the last 1
nothing left to move: that is the pose

A candidate says: scene points $s_1, s_2$ are mesh points $m_1, m_2$. A pose is 6 numbers (3 position, 3 rotation). Moving the mesh to agree with that match uses all 6: $3 + 2 + 1$.

- A **wrong** candidate gives a wrong pose just as easily, so no single match is trusted.
- **Vote:** every candidate from every scene pair adds one vote to the pose it gives. Wrong ones scatter; right ones pile up on the same pose. Most votes wins.

# ICP: matches from a rough guess




1. Draw the model at a rough pose guess.
2. Match each scene point to the **nearest** model point (top).
3. Procrustes on those pairs: a better pose. Repeat until it settles. This is ICP, iterative closest point (L07).

Bottom: mismatch after ICP vs how far off the start was. Close start: nearest points are mostly right, and it converges. Far start: they are wrong, and it stops in a wrong dip.

**So: a global guess first (votes, or a network), then ICP to polish.**

# Learned pose, in three generations

| | | what you must have |
|---|---|---|
| instance-level | PoseCNN, DenseFusion (2018-19) | the exact objects, at training |
| category-level | NOCS (2019) | objects of a known category |
| novel object | MegaPose (2022), FoundationPose (2024) | a CAD model or a few photos, at test time |

Each generation needs less at test time. The third: **render and compare**. Draw the object at the guess; a network says how to move it.

# Category level: one coordinate frame for all mugs

*NOCS, Wang et al., CVPR 2019*

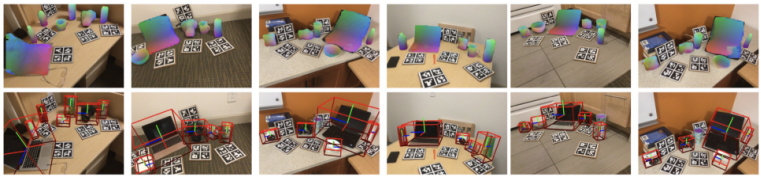
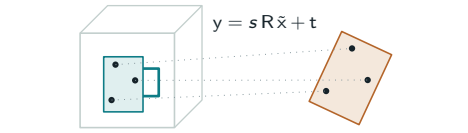


Figure: NOCS (Wang et al., CVPR 2019), MIT licence

- **Normalised object coordinate space**: every mug upright, handle aligned, centred, box diagonal 1.
- A network predicts each pixel's NOCS coordinate (the colours). Same part of any mug, same colour.
- Needs the category, not a CAD model of *this* mug.

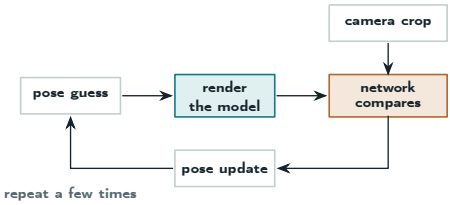
# From NOCS to pose and size




Each pixel matches predicted $\tilde{x}$ (cube) to measured y (depth). Fit rotation, translation and scale $s$ with RANSAC: Procrustes plus scale (Umeyama).

**Two pixels at $\tilde{x} = (\pm 0.1, 0, 0)$ are measured 6 cm apart. The mug's NOCS box is $0.48 \times 0.6 \times 0.64$. Scale and size?**

# Render and compare




- The network learns to compare a rendering with a photo of the same object.
- So it works on any object you can render.
- Same idea as Gaussian Splatting SLAM tracking (L09), with the object moving instead of the camera.

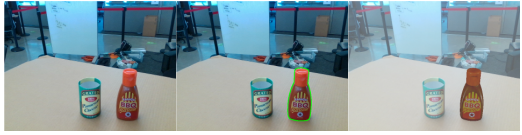


Figure: MegaPose (Labbé et al., CoRL 2022), Apache-2.0

# MegaPose: learning to compare on 2 million renders

*Labbé et al., CoRL 2022*

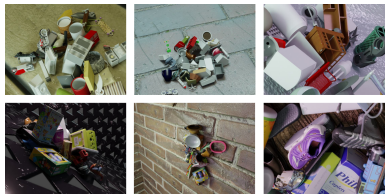
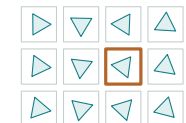


Figure: MegaPose (Labbé et al., CoRL 2022), Apache-2.0

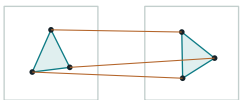
- 2 million renders of 20,000+ objects (Google Scanned Objects, ShapeNet) in random piles.
- Coarse: score renders at many poses by *could the refiner fix this?* Keep the best.
- Refine: a few render-and-compare steps. FoundationPose keeps both stages.

# GigaPose: fewer templates

*Nguyen et al., CVPR 2024*



match the view: 2 numbers



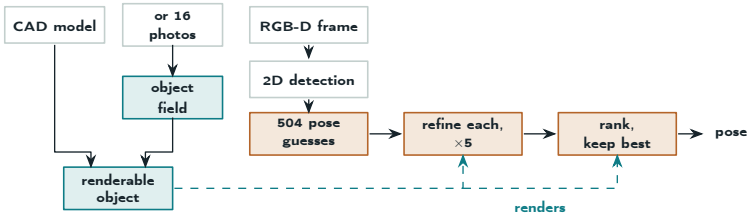
patch matches: in-plane turn, scale and 2D shift, 4 numbers

**With FoundationPose's sampling, how many templates does each need?**

- Only the viewing direction (2 of rotation's 3 numbers) changes appearance; in-plane turns and zoom do not.
- Templates over viewing direction only; in-plane rotation, scale and shift from patch matches.
- RGB only; 35× faster than the previous best.

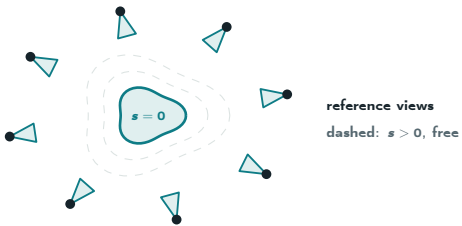
# FoundationPose: one network, two setups

*Wen, Yang, Kautz and Birchfield, CVPR 2024*



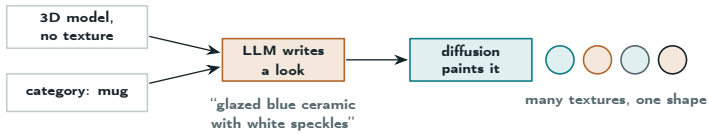

- Model-based: render the CAD model. Model-free: render a neural object field built from 16 photos.
- Then the same steps (guess, refine, rank) and the same weights for both, and for tracking.

# Model-free: an object from 16 photos




- Two small networks (L06): signed distance $s(x)$ at point x for shape, plus colour.
- Fit by volume rendering to the RGB-D photos; depth supervises $s$ near the surface. Seconds, once per object.
- Marching cubes on $s = 0$ gives a textured mesh, rendered like a CAD model.

# Training data nobody labelled




- 40,000+ Objaverse shapes (1,156 categories) plus Google Scanned Objects: 1.2 million rendered images.
- An LLM describes a plausible look per category; TexFusion paints it on.
- Beats pasting random ImageNet photos as textures.

# Finding the object first

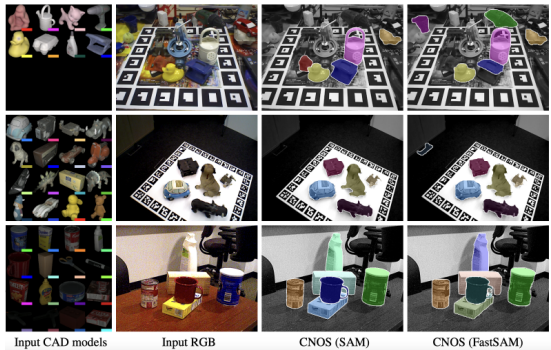
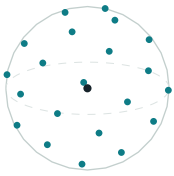



Figure: CNOS (Nguyen et al., ICCVW 2023), MIT licence

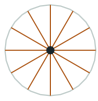
FoundationPose needs a 2D box first. **CNOS** uses L11's tools: SAM proposes masks; DINOv2 features match them to renders of the model.

# 504 guesses



42 viewpoints

$\times$



12 in-plane turns

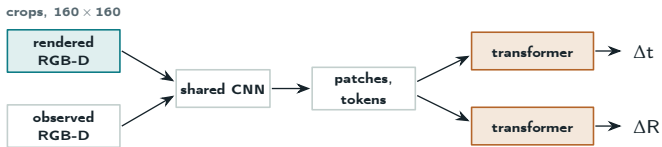
$= 504$ guesses

- Rotation: 42 icosphere viewpoints $\times$ 12 in-plane turns.
- Translation: one guess, at the box's median depth.

**How many guesses, and how many refiner passes at 5 iterations each?**

# The refiner

*render at the guess, crop, compare*




- Both crops follow the *guessed* pose, resized to $160 \times 160$: the object always fills the crop the same way.
- Shared CNN, then transformer (L10). Outputs: translation correction $\Delta t \in \mathbb{R}^3$, rotation correction $\Delta R$; L2 loss.

# Applying the update: turn about the object



about the camera



about the object's centre

$$t^{+} = t + \Delta t, \qquad R^{+} = \Delta R\, R$$

**Object 80 cm in front of the camera. Left-multiply the whole pose by a $10^{\circ}$ turn about the camera's $x$ axis instead. How far does its centre move?**

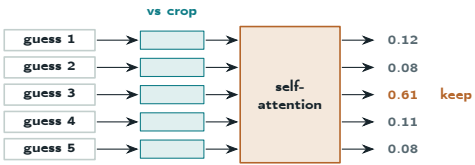
# The same update, in this course's convention

FoundationPose's $\Delta \mathrm{R} = \mathrm{Exp}(\boldsymbol{\omega})$ is about **camera** axes, applied on the left. This course perturbs on the **right**, about the object's own axes (L03).

**Rewrite $\Delta \mathrm{R}\,\mathrm{R}$ as $\mathrm{R}$ times something. For $\mathrm{R} = \mathrm{R}_z(90^\circ)$, which object axis is camera $x$?**

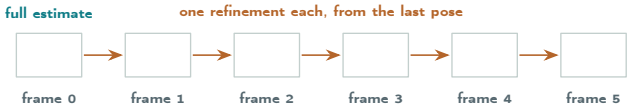
*Either convention works. Check which one a paper uses.*

# Picking the winner: two-level ranking




- Level 1: compare each refined guess's render with the crop; one 512-D embedding each.
- Level 2: self-attention across guesses, so scores are relative.
- Triplet loss: the guess nearest the truth must win by a margin.

# Tracking: one step per frame




- After frame one: no guesses, no ranking. Refine once from the last pose.
- Full estimate about 1.3 s; tracking about 32 Hz.
- Lost track: run the full estimate again.

# Tracking with no model at all: BundleTrack

*Wen and Bekris, IROS 2021*

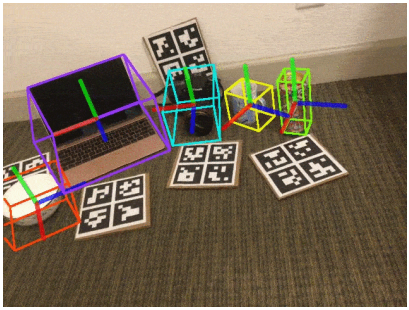
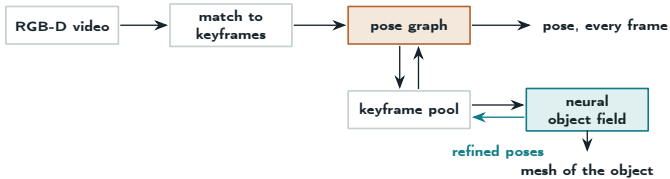


Figure: BundleTrack (Wen and Bekris, IROS 2021), Apache-2.0

- Input: an RGB-D video and a first-frame mask. Nothing else.
- Match against a **memory of keyframes** and solve a small pose graph (L04): bounded drift.
- 10 Hz. Each box is a tracked object.

# BundleSDF: track it and build it at once

*Wen et al., CVPR 2023*




- BundleTrack, plus a neural object field (SDF and colour) trained on the keyframes in the background.
- The field refines the poses, and yields a mesh of an unscanned object.
- FoundationPose builds the same kind of field offline, from 16 posed photos.

# Scoring a pose: ADD

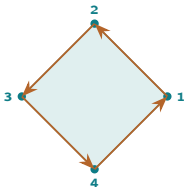
Model points $x \in \mathcal{M}$; true pose $(R, t)$, estimate $(\hat{R}, \hat{t})$. Average the gap:

$$\mathrm{ADD} = \frac{1}{|\mathcal{M}|} \sum_{x \in \mathcal{M}} \left\| (Rx + t) - (\hat{R}x + \hat{t}) \right\|$$

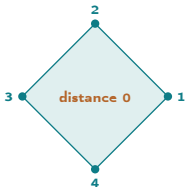
- In metres: a rotation error counts more at points far from the centre.
- Reported as % of poses under $0.1\times$ diameter, or area under the accuracy curve to 10 cm.

# Symmetric objects break ADD

$$\text{ADD-S} = \frac{1}{|\mathcal{M}|} \sum_{x_1 \in \mathcal{M}} \min_{x_2 \in \mathcal{M}} \left\| (R x_1 + t) - (\hat{R} x_2 + \hat{t}) \right\|$$




ADD: point to its own twin




ADD-S: point to the nearest

**Corners 5 cm from the centre; the estimate is turned 90°. ADD, ADD-S, and do they pass at $0.1d$?**

# BOP: how the field keeps score

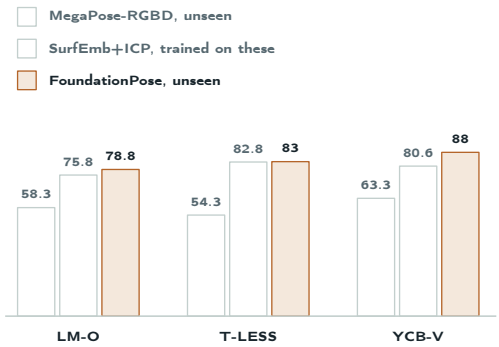
*Hodaň et al., ECCV 2018, and yearly challenges*

| Error | What it measures, and who cares |
|---|---|
| VSD | Visible-surface disagreement. Poses that look alike count as equal. |
| MSSD | Max 3D surface distance, over symmetries. Matters for grasping. |
| MSPD | Max image distance, over symmetries. Matters for AR. |
| AR | Recall over a range of thresholds, averaged over the three. |

Fixed datasets, a public leaderboard, unseen-object tracks since 2023. FoundationPose ranked first there when submitted.

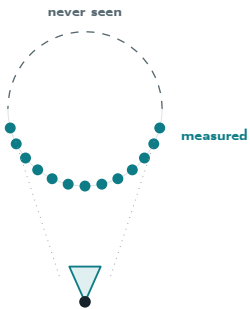
# What FoundationPose reports

*BOP average recall, model-based, from the paper's Table 3*



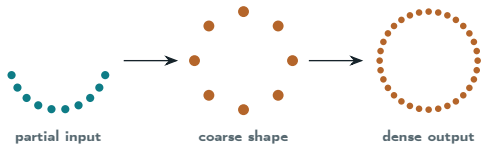

- Beats MegaPose by 20 to 29; edges SurfEmb, which **trained on these objects**.
- Model-free YCB-Video, ADD AUC: 91.5 vs 42.1 (FS6D).
- Tracking YCBInEOAT, ADD AUC: 93.1 vs 87.0 (BundleSDF).
- Ablation, ADD lost: 2.5 without ranking level 2, 0.7 without LLM textures.

# One view sees half an object




- Depth sees the near side only. Back, inside, occluded parts: unknown.
- With a model, pose fills the gap. Without one, the planner sees a shell.
- Fusion (L06) helps only if the robot moves around it.

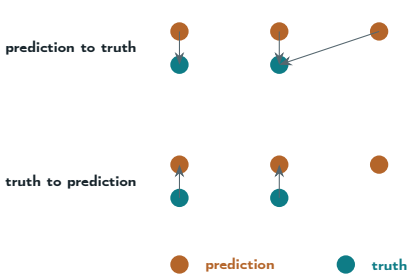
# Shape completion




- Learn a shape prior; predict the whole from the part. **PCN** (2018): coarse points, then refine. **PoinTr** (2021): transformer, point set to point set.
- Completing before planning raises grasp success on unseen objects (Varley et al., 2017).
- The hidden side is a *guess*: trust it less than measurements.

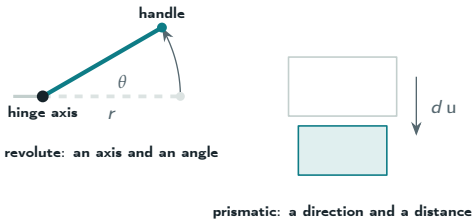
# Scoring a completion: Chamfer distance

$$\mathrm{CD}(P, G) = \frac{1}{|P|} \sum_{\mathrm{p} \in P} \min_{\mathrm{g} \in G} \|\mathrm{p} - \mathrm{g}\|^2 + \frac{1}{|G|} \sum_{\mathrm{g} \in G} \min_{\mathrm{p} \in P} \|\mathrm{g} - \mathrm{p}\|^2$$




**Prediction $P$: $(0,0), (10,0), (20,0)$ cm; truth $G$: $(0,0), (10,0)$. Each term, then $P = \{(0,0)\}$.**

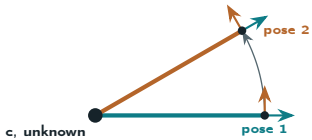
# Objects that move: articulation




- **Revolute:** axis a through point c, angle $\theta$: $\mathrm{p}' = \mathrm{R_a}(\theta)(\mathrm{p} - \mathrm{c}) + \mathrm{c}$.
- **Prismatic:** direction u, distance $d$: $\mathrm{p}' = \mathrm{p} + d\,\mathrm{u}$.

**A handle 0.7 m from the hinge; the door opens 30°. How far does the hand travel, and how far apart are start and end?**

# Finding the hinge from two poses




Two tracked door poses give a relative motion $p \mapsto Rp + t$. The hinge c is the point it leaves fixed.

**$\operatorname{tr} R = 2.732$ and $t = (0.384, -0.433)$ m in the floor plane. Angle, then hinge c.**

# Learning a joint by watching it move

*Ditto, Jiang, Hsu and Zhu, CVPR 2022*

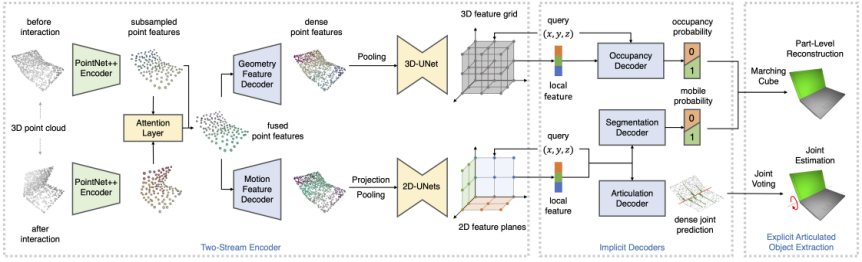


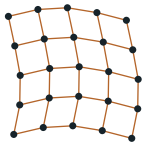

Figure: Ditto (Jiang et al., CVPR 2022), MIT licence

- In: point clouds before and after someone (or the robot) moves a part.
- Out: the moving part, joint type and axis, a mesh per part.

# Deformable objects



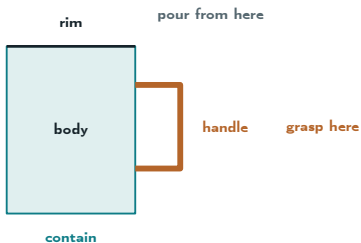
rope: a chain of points



cloth: a mesh of points

- Rope, cloth, food, cables: no 6D pose. The state is every point.
- Keypoints along a rope; mesh or particles for cloth; tracked per frame.
- A folded towel hides most of itself.

# Parts and affordances




- Pose says where the mug is, not **which part to use**.
- Parts: PartNet labels shapes into a part hierarchy.
- Affordances: per-point scores for *grasp here*, *push here*; learned from interaction (Where2Act) or language and images.
- L11's open-vocabulary features query parts by name: *the handle*.

# Where it fails

- Detection: wrong or missing box, meaningless pose.
- Depth: shiny, transparent, thin objects.
- Texture-less and occluded: nothing pins the rotation (the paper's own failure case).
- Symmetry: a perfect score, but the handle on the wrong side.
- Model-free: needs 16 posed views, all sides, first.

# Reading FoundationPose

*for Sunday's response*

- Claim: one network, no fine-tuning, model-based and model-free, plus tracking.
- Evidence: comparisons with single-setting methods; ablations of LLM textures and ranking level 2.
- Assumptions to question: a good 2D box; reliable depth; 16 posed reference views.
- Next experiment: remove the detector, or the depth.

# Learn it again, differently

*other explanations*

- **Tedrake, *Robotic Manipulation*, ch. 4** (manipulation.mit.edu/pose.html): Procrustes, ICP minima, interactive plots.
- **Open3D: global registration, then ICP** (open3d.org/docs): FPFH + RANSAC, then ICP, in a dozen lines.
- **BOP site** (bop.felk.cvut.cz): datasets and exact VSD, MSSD, MSPD.
- **FoundationPose page** (nvlabs.github.io/FoundationPose): videos, failures included.

# What you should be able to do

- Chain $T_{wc}T_{co}T_{og}$ to place a grasp.
- Say what instance, category and novel-object pose estimation are each given.
- Solve a 2D Procrustes problem by hand, and say why ICP needs a global guess.
- Walk through FoundationPose: object field, synthetic data, 504 guesses, refiner, ranking, tracking.
- Explain why its rotation update is about the object's centre, and convert it to a right perturbation.
- Compute ADD and ADD-S for a symmetric object, and name BOP's three errors.
- Write the motion of a point under a revolute and a prismatic joint.

# Next

*fall break, then scene graphs*

- **Responses due Sun Oct 18:** OpenScene (L11) and FoundationPose (today).
- **No class Mon Oct 19:** fall study break.
- **L13 (Wed Oct 21):** open-set 3D scene graphs, queried by a language model. Response paper: ConceptGraphs.
- **Project proposals due Fri Oct 23:** a target paper and an evaluation.