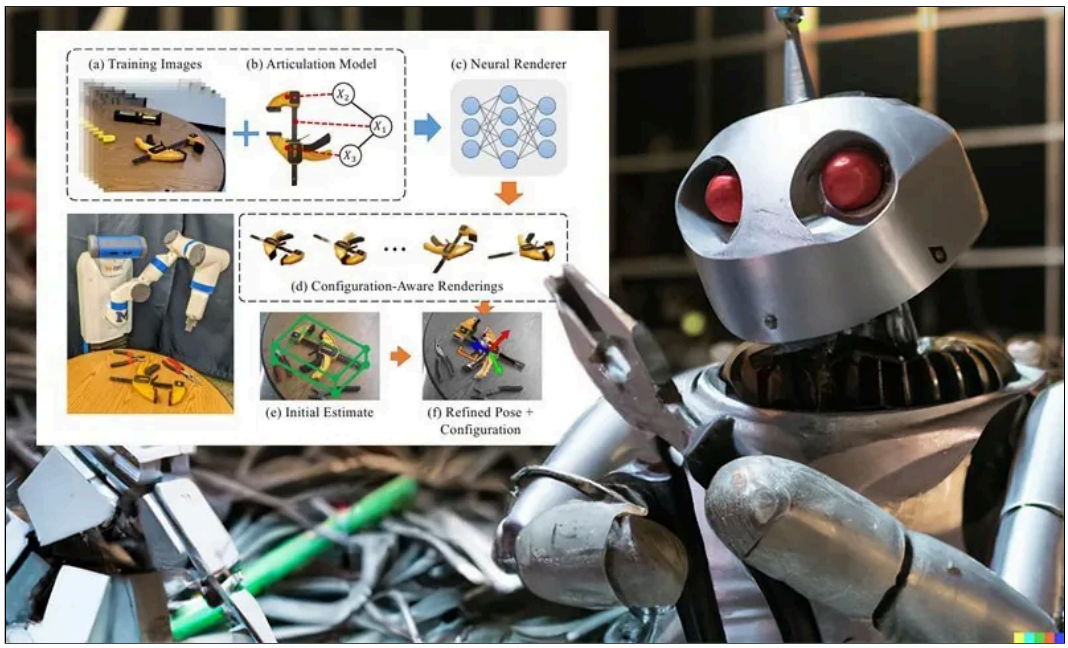
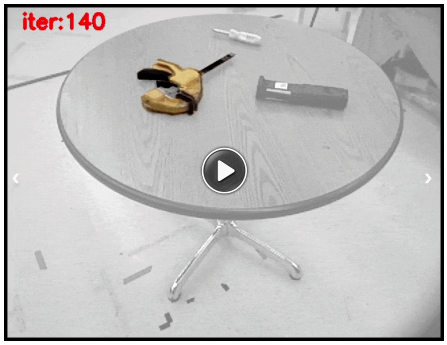
Teaching Robots About Tools With Neural Radiance Fields (NeRF)

Originally published at https://www.unite.ai/teaching-robots-about-tools-with-neural-radiance-fields-nerf/



New research from the University of Michigan proffers a way for robots to understand the mechanisms of tools, and other real-world articulated objects, by creating Neural Radiance Fields (NeRF) objects that demonstrate the way these objects move, potentially allowing the robot to interact with them and use them without tedious dedicated preconfiguration.



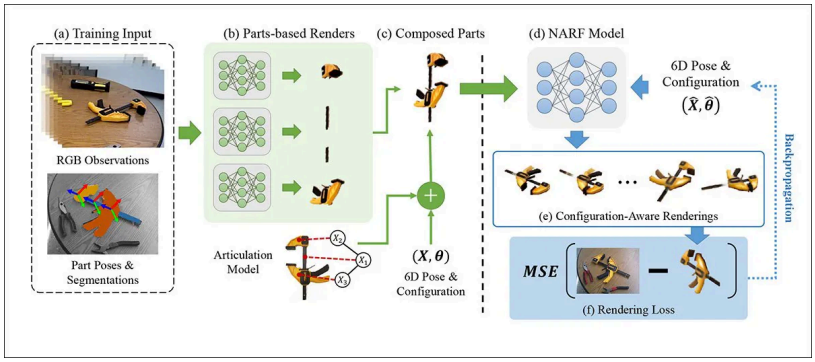
CLICK TO VIEW ANIMATED GIF

By utilizing known source references for the internal motility of tools (or any object with a suitable reference), NARF22 can synthesize a photorealistic approximation of the tool and its range of movement and type of operation. Source: https://progress.eecs.umich.edu/projects/narf/

Robots that are required to do more than avoid pedestrians or perform elaborately pre-programmed routines (for which non-reusable datasets have probably been labeled and trained at some expense) need this kind of adaptive capacity if they are to work with the same materials and objects that the rest of us must contend with.

To date, there have been a number of obstacles to imbuing robotic systems with this kind of versatility. These include the paucity of applicable datasets, many of which feature a very limited number of objects; the sheer expense involved in generating the kind of photorealistic, mesh-based 3D models that can help robots to learn instrumentality in the context of the real world; and the non-photorealistic quality of such datasets as may actually be suitable for the challenge, causing the objects to appear disjointed from what the robot perceives in the world around it, and training it to seek a cartoon-like object that will never appear in reality.

To address this, the Michigan researchers, whose paper is titled *NARF22: Neural Articulated Radiance Fields for Configuration-Aware Rendering*, have developed a two-stage pipeline for generating NeRF-based articulated objects which have a 'real world' appearance, and which incorporate the movement and ensuing limitations of any particular articulated object.



Though it appears more complex, the essential two stages of the NARF22 pipeline involve rendering static parts of motile tools, and then compositing these eleme is informed about the parameters of movement that these parts have, relative to each other. Source: https://arxiv.org/pdf/2210.01166.pdf

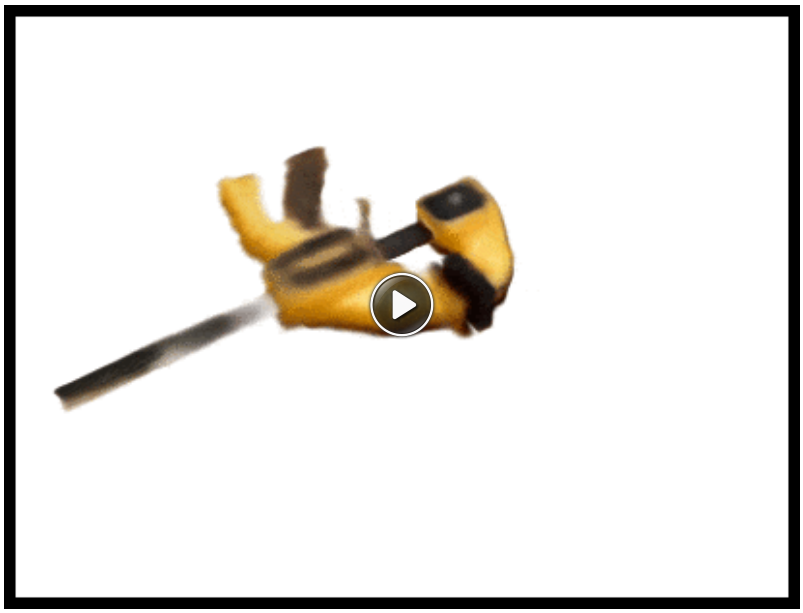
The system is called *Neural Articulated Radiance Field* – or NARF22, to distinguish it from another similarly-named project.

NARF22

Determining whether or not an unknown object is potentially articulated requires an almost inconceivable amount of human-style prior knowledge. For instance, if you had never seen a closed drawer before, it might appear to be any other kind of decorative paneling – it's not until you've actually opened one that you internalize 'drawer' as an articulated object with a single axis of movement (forward and backward).

Therefore NARF22 is not intended as an exploratory system for picking things up and seeing if they have actionable moving parts – almost simian behavior which would entail a number of potentially disastrous scenarios. Rather, the framework is predicated on knowledge available in Universal Robot Description Format (URDF) – an open source XML-based format that's widely applicable and suitable for the task. A URDF file will contain the usable parameters of movement in an object, as well as descriptions and other labeled facets of the parts of the object.

In conventional pipelines, it's necessary to essentially describe the articulation capabilities of an object, and to label the pertinent joint values. This is not a cheap or easily-scalable task. Instead, the NaRF22 workflow renders the individual components of the object before 'assembling' each static component into an articulated NeRF-based representation, with knowledge of the movement parameters provided by URDF.



CLICK TO VIEW ANIMATED GIF

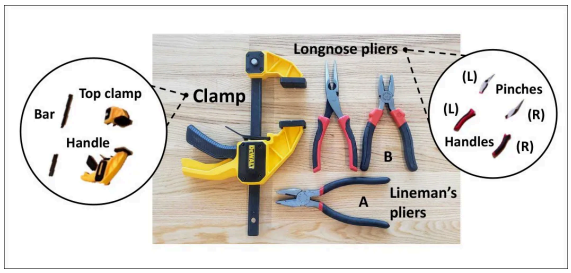
In the second stage of the process, an entirely new renderer is created which incorporates all the parts. Though it might be easier to simply concatenate the individual parts at an earlier stage and skip this subsequent step, the researchers observe that the final model – which was trained on a NVIDIA RTX 3080 GPU under an AMD 5600X CPU – has lower computational demands during backpropagation than such an abrupt and premature assembly.

Additionally, the second-stage model runs at twice the speed of a concatenated, 'brute-forced' assembly, and any secondary applications which may need to utilize information about static parts of the model will not need their own access to URDF information, because this has already been incorporated into the final-stage renderer.

Data and Experiments

The researchers conducted a number of experiments to test NARF22: one to evaluate qualitative rendering for each object's configuration and pose; a quantitative test to compare the rendered results to similar viewpoints seen by real-world robots; and a demonstration of the configuration estimation and a 6 DOF (depth of field) refinement challenge that used NARF22 to perform gradient-based optimization.

The training data was taken from the Progress Tools dataset from an earlier paper by several of the current work's authors. Progress Tools contains around six thousand RGB-D (i.e., including depth information, essential for robotics vision) images at 640×480 resolution. Scenes used included eight hand tools, divided into their constituent parts, complete with mesh models and information on the objects' kinematic properties (i.e., the way they are designed to move, and the parameters of that movement).



The Progress Tools dataset features four articulated tools. The images above are NeRF-based renders from NARF22.

For this experiment, a final configurable model was trained using only linesmen's pliers, longnose pliers, and a clamp (see image above). The training data contained a single configuration of the clamp, and one for each of the pliers.

The implementation of NARF22 is based on FastNeRF, with the input parameters modified to concentrate on concatenated and spatially-encoded pose of the tools. FastNeRF uses factorized multilayer perceptron (MLP) paired with a voxelized sampling mechanism (voxels are essentially pixels, but with full 3D coordinates, so that they can operate in a three-dimensional space).

For the qualitative test, the researchers observe that there are several occluded parts of the clamp (i.e., the central spine, that cannot be known or guessed by observing the object, but only by interacting with it, and that the system has difficulty creating this 'unknown' geometry.

Qualitative renderings of tools.

By contrast, the pliers were able to generalize well to novel configurations (i.e. to extensions and movements of their parts which are within the URDF parameters, but which are not explicitly addressed in the training material for the model.

The researchers observe, however, that labeling errors for the pliers led to a diminution of rendering quality for the very detailed tips of the tools, negatively affecting the renderings – a problem related to much wider concerns around labeling logistics, budgeting and accuracy in the computer vision research sector, rather than any procedural shortcoming in the NARF22 pipeline.

Tool	Dataset	Per Pixel MSE	N (instances)
Clamp	Test	0.03343	272
	Train	0.03234	2483
	Novel Config	0.13245	200
Lineman's (A)	Test	0.02991	171
	Train	0.02941	1557
	Novel Config	0.10136	345
Lineman's (B)	Test	0.06348	171
	Train	0.06318	1557
	Novel Config	0.12500	326
Longnose	Test	0.08045	171
	Train	0.07900	1557
	Novel Config	0.10241	171

Results from the render accuracy test.

For the configuration estimation tests, the researchers performed pose refinement and configuration estimation from an initial 'rigid' pose, avoiding any of the caching or other accelerative workarounds used by FastNeRF itself.

They then trained 17 well-ordered scenes from the test set of Progress Tools (which had been held aside during training), running through 150 iterations of gradient descent optimization under the Adam optimizer. This procedure recovered the configuration estimation 'extremely well', according to the researchers.

Metric	Mean	Std. Dev.
ADD (m)	0.0107	0.0043
Configuration Err. (m)	0.0073	0.0065

Results from the configuration estimation test.

First published 5th October 2022.

Writer on machine learning, domain specialist in human image synthesis. Former head of research content at Metaphysic.ai, until its dissolution into DNEG's Brahma.ai.
Portfolio site: martinanderson.ai
Contact: martin@martinanderson.ai

Discover More