CS 280 Computer Vision
Spring 2018 Tulsiani Lecture 23

1 Lecture

Single-View 3D Reconstruction

The goal is for the computer to infer the 3D structure of a scene from a single 2D image. Specific tasks include relative distance estimation, path planning, and shape description – even for parts of the scene without actual pixels (e.g. occluded regions). All of these tasks, done in their entirety, require a 3D understanding.

The difference between this problem and "3D from correspondence" methods (SfM, multi-view stereo) is that "3D from correspondence" methods require multiple images to build 3D models.

$$\text{single RGB image} \rightarrow \boxed{\text{single-view 3D reconstruction}} \rightarrow \text{3D representation}$$

3D representations vary:

- **Depth image:** how far away each pixel is. Drawback: doesn't capture unseen parts of the scene.
- **Scene layout:** coarsely captures geometry of scene (e.g. planes for walls, blocks for objects).
- **Voxels (binary 3D mask):** volumetric 3D representation of an object. A binary 3D image: *1 for "cell occupied," 0 for "cell not occupied."*
- **CAD models:** a 3D model, e.g. the closest match on 3D Warehouse.
- **Deformable shape models:** one of a collection of models whose shapes can vary in certain ways (thinner chair legs, fatter stem on bike, etc.). Relies on possession of a rough mean shape and knowledge of how the shape can deform.
- **Primitives, point clouds, octrees...**

Specific choice of representation is driven by (a) what we can predict and (b) what we want to do on top of the prediction.

Note: given a 2D projection, there are countless 3D scenes that could have given rise to the projection. So mathematically, SV3DR is an ill-posed and underspecified task. But our world is structured. Having lived in the world, we have some prior knowledge of how scenes can be put together; only one or a few of the infinite depth interpretations should be feasible. We can ***leverage the world's regularities****.*

- These regularities can either be given (explicitly, as a prior model) or learned from training data.

Explicit modeling is problematic because it's nigh-impossible to capture all possible variations of scenes, and it's typically not robust because it relies on low level processing steps (e.g. matching based on edge or block representations) for which errors have huge high-level ramifications.

Instead, we'll usually want to learn an implicit model from lots and lots of training data.

Learning Using Direct Supervision

This is the implicit case, where we pass an RGB image through our learned predictor and receive our desired 3D representation as output. The full process:

1. Collect training data (images with corresponding 3D representations).

2. Learn a predictor using this training data.
3. Use the predictor.

For example, if predicting depth images, we can use Kinect-esque depth cameras to collect that data.

We can learn to **reconstruct a scene** by training a model which outputs (a) individual object data and (b) the scene layout if there were no objects.

Geometric Consistency

A problem: we want to reconstruct real-world scenes, but all we have is synthetic training data. And it's really hard to create 3D annotations for real-world imagery. *A solution*: learn via **geometric consistency**. Take multiple photos of a 3D scene from different perspectives, and predict the scene from one image. Then make sure the predicted model of the scene consistently gives rise (via projections) to all of the other views. Brilliant!

Requires no ground-truth 3D at training time – only a few views of many different scenes.

Similarly, in *learning to predict depth* we can track points from the perspective of a moving vehicle, predict depth from the view at one time step, and make sure that this depth map (when used to project points) is consistent with points in views from different time steps.

Even if we don't have multiple views, we can check consistency in that the predicted 3D scene should project back onto the original RGB image.