# 6.801/6.866: Machine Vision, Lecture 13

Professor Berthold Horn, Ryan Sander, Tadayuki Yoshitake
MIT Department of Electrical Engineering and Computer Science
Fall 2020

These lecture summaries are designed to be a review of the lecture. Though I do my best to include all main topics from the lecture, the lectures will have more elaborated explanations than these notes. Therefore, if you're looking for the most rigorous review and treatment of these topics, we encourage you to rewatch the lecture videos. With that said, we hope these summaries are beneficial for your learning. If you have any feedback for these lecture summaries, please submit it **here**.

## 1 Lecture 13: Object Detection, Recognition and Pose Determination, PatQuick (US 7,016,539)

In this lecture, we will begin by looking at some general problems for object detection and pose estimation of objects in an image, and also look at some optimization algorithms we can use for finding optimal matches between a "master image" / template image, which is the object we look for, and this object in another image (perhaps in a scene). We then look at a patent describing some of the approaches taken to solve some of these aforementioned tasks.

### 1.1 Motivation & Preliminaries for Object Detection/Pose Estimation

Object detection and pose estimation builds on top of previous machine vision tasks we have covered. Some specific tasks include:

- Object detection - i.e. detect and locate object in an image
- Recognize object
- Determine pose of detected/recognized object
- Inspect object

**Motivation for these approaches**: In machine vision problems, we often manipulate objects in the world, and we want to know what and where these objects are in the world. In the case of these specific problems, we assume prior knowledge of the precise edge points of these objects (which, as we discussed in the two previous lectures, we know how to compute!)

#### 1.1.1 "Blob Analysis"/"Binary Image Processing"

We can use thresholding and other algorithms such as finding a convex hull to compute elements of an object in a binary image (black/white) such as:

- Area (moment order 0)
- Perimeter
- Centroid (moment order 1)
- "Shape" (generalization of different-order moments)
- Euler number - In this case this is the number of blobs minus number of holes

A few notes about computing these different quantities of interest:

- We have seen from previous lectures that we can compute some of these elements, such as perimeter, using **Green's Theorem**. We can also accomplish this with **counting** - can be done simply by counting pixels based off of whether their pixel is a 0 or 1.

- However, the issue with these approaches is that they require **thresholding** - i.e. removing points from any further consideration early on the process and possibly without all information available; essentially removing potentially viable points too early.
- **Shape**: As introduced above, shape is often computed by computing moments of any order. Recall the definition of moments of a 2D shape:
  1. **O-order**: $\iint_D E(x,y)dxdy \rightarrow$ **Area**
  2. **1-order**: $\iint_D E(x,y)xydxdy \rightarrow$ **Centroid**
  3. **2-order**: $\iint_D E(x,y)x^2y^2dxdy \rightarrow$ **Dispersion**

  $\vdots$
  4. **k-order**: $\iint_D E(x,y)x^ky^kdxdy$
- Note that these methods are oftentimes applied to **processed**, not **raw** images.

### 1.1.2 Binary Template

We will discuss this more later, but this is crucial for the patent on object detection and pose estimation that we will be discussing today. A binary template is:

- A "master image" to define the object of interest that we are trying to detect/estimate the pose of
- You will have a template of an object that you can recognize the object and get the pose/attitude.

### 1.1.3 Normalized Correlation

This methodology also plays a key role in the patent below.

**Idea**: Try all possible positions/configurations of the pose space to create a match between the template and runtime image of the object. If we are interested in the squared distance between the displaced template and the image in the other object (for computational and analytic simplicity, let us only consider rotation for now), then we have the following optimization problem:

$$\min_{\delta_x,\delta_y} \iint_D (E_1(x-\delta_x, y-\delta_y) - E_2(x,y))^2 dxdy$$

Where we have denoted the two images separately as $E_1$ and $E_2$.

In addition to framing this optimization mathematically as minimizing the squared distance between the two images, we can also conceptualize this as maximizing the correlation between the displaced image and the other image:

$$\max_{\delta_x,\delta_y} \iint_D E_1(x-\delta_x, y-\delta_y)E_2(x,y)dxdy$$

We can prove mathematically that the two are equivalent. Writing out the first objective as $J(\delta_x, \delta_y)$ and expanding it:

$$\begin{aligned} J(\delta_x,\delta_y) &= \iint_D (E(x-\delta_x, y-\delta_y) - E_2(x,y))^2 dxdy \\ &= \iint_D (E_1^2(x-\delta_x, y-\delta_y) - 2\iint_D E_1(x-\delta_x, y-\delta_y)E_2(x,y)dxdy + \iint_D E_2^2(x,y)dxdy \\ &\implies \arg\min_{\delta_x,\delta_y} J(\delta_x,\delta_y) = \arg\max_{\delta_x,\delta_y} \iint_D E_1(x-\delta_x, y-\delta_y)E_2(x,y)dxdy \end{aligned}$$

Since the first and third terms are constant, and since we are minimizing the negative of a scaled correlation objective, this is equivalent to maximizing the correlation of the second objective.

We can also relate this to some of the other gradient-based optimization methods we have seen using Taylor Series. Suppose $\delta_x, \delta_y$ are small. Then the Taylor Series Expansion of first objective gives:

$$\iint_D (E_1(x-\delta_x, y-\delta_y) - E_2(x,y))^2 dxdy = \iint_D (E_1(x,y) - \delta_x\frac{\partial E_1}{\partial x} - \frac{\partial E_1}{\partial y} + \cdots - E_2(x,y))^2 dxdy$$

If we now consider that we are looking between consecutive frames with time period $\delta_t$, then the optimization problem becomes (after simplifying out $E_1(x,y) - E_2(x,y) = -\delta_t \frac{\partial E}{\partial t}$):

$$\min_{\delta_x, \delta_y} \iint_D (-\delta_x E_x - \delta_y E_y - \delta_t E_t)^2 dxdy$$

Dividing through by $\delta_t$ and taking $\lim_{\delta_t \to 0}$ gives:

$$\min_{\delta_x, \delta_y} \iint_D (uE_x + vE_y + E_t)^2 dxdy$$

A few notes about the methods here and the ones above as well:

- Note that the term under the square directly above looks similar to our BCCE constraint from optical flow!
- Gradient-based methods are cheaper to compute but only function well for small deviations $\delta_x, \delta_y$.
- Correlation methods are advantageous over least-squares methods when we have scaling between the images (e.g. due to optical setting differences): $E_1(x,y) = kE_2(x,y)$ for some $k \in \mathbb{R}$ .

Another question that comes up from this: How can we match at different contrast levels? We can do so with **normalized correlation**. Below, we discuss each of the elements we account for and the associated mathematical transformations:

1. **Offset**: We account for this by subtracting the mean from each brightness function:

$$E_1^{'}(x,y) = E_1(x,y) - \bar{E}_1, \quad \bar{E}_1 = \frac{\iint_D E_1(x,y)dxdy}{\iint_D dxdy}$$
$$E_2^{'}(x,y) = E_2(x,y) - \bar{E}_2, \quad \bar{E}_2 = \frac{\iint_D E_2(x,y)dxdy}{\iint_D dxdy}$$

   This removes offset from images that could be caused by changes to optical setup.

2. **Contrast**: We account for this by computing normalized correlation, which in this case is the Pearson correlation coefficient:

$$\frac{\iint_D E_1^{'}(x-\delta_x, y-\delta_y) E_2^{'}(x,y) dxdy}{\left(\sqrt{\iint_D E_1^{'}(x-\delta_x, y-\delta_y)dxdy}\right)\left(\sqrt{\iint_D E_2^{'}(x,y)dxdy}\right)} \in [-1,1]$$

   Where a correlation coefficient of 1 denotes a perfect match, and a correlation coefficient of -1 denotes a perfectly imperfect match.

**Are there any issues with this approach?** If parts/whole images of objects are obscured, this will greatly affect correlation computations at these points, even with proper normalization and offsetting.

With these preliminaries set up, we are now ready to move into a case study: a patent for object detection and pose estimation using probe points and template images.

## 1.2 Patent 7,016,539: Method for Fast, Robust, Multidimensional Pattern Recognition

This patent aims to extend beyond our current framework since the described methodology can account for more than just translation, e.g. can account for:

- Rotation
- Scaling
- Shearing

### 1.2.1 Patent Overview

This patent describes a patent for determining the presence/absence of a pre-determined pattern in an image, and for determining the location of each found instance within a multidimensional space.

A diagram of the system can be found below: A few notes about the diagram/aggregate system:

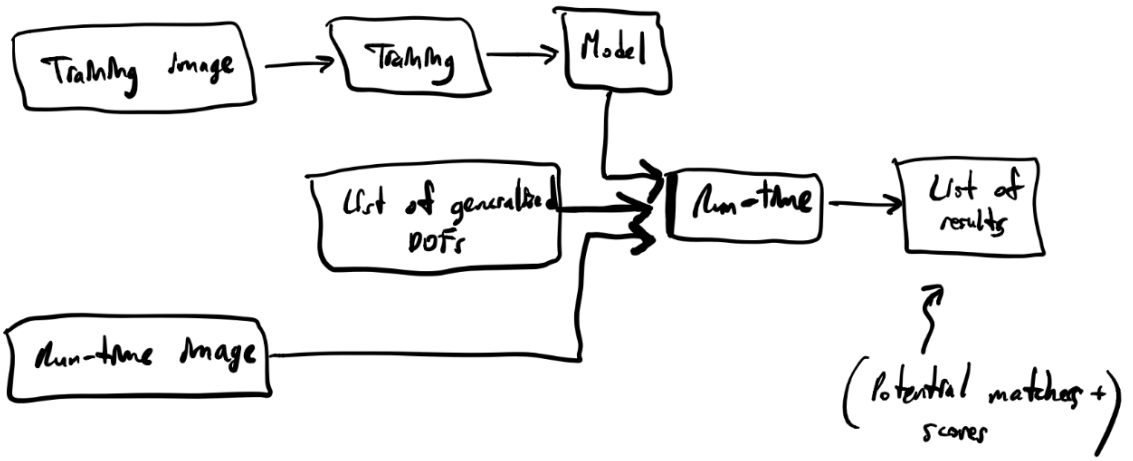


Figure 1: System diagram.

- A **match score** is computed for each configuration, and later compared with a threshold downstream. This process leads to the construction of a high-dimensional matches surface.
- We can also see in the detailed block diagram from this patent document that we greatly leverage gradient estimation techniques from the previous patent on fast and accurate edge detection.
- For generalizability, we can run this at multiple scales/levels of resolution.

#### 1.2.2 High-level Steps of Algorithm

1. Choose appropriate level of **granularity** (defined as "selectable size below which spatial variations in image brightness are increasingly attenuated, and below which therefore image features increasingly cannot be resolved") and store in model.
2. Process training/template image to obtain boundary points.
3. Connect neighboring boundary points that have consistent directions.
4. Organize connected boundary points into chains.
5. Remove short or weak chains.
6. Divide chains into segments of low curvature separated by conrner of high curvature.
7. Create evenly-spaced along segments and store them in model.
8. Determine pattern contrast and store in model.

A few notes about this procedure:

- To increase the efficiency of this approach, for storing the model, we store **probes** (along with granularity and contrast), not the image for which we have computed these quantities over.
- When comparing gradients between runtime and training images, project probe points onto the other image - we do not have to look at all points in image; rather, we **compare gradients** (direction and magnitude - note that magnitude is often less viable/robust to use than gradient direction) between the training and runtime images only at the probing points.
- We can also weight our probing points, either automatically or manually. Essentially, this states that some probes are more important that others when scoring functions are called on object configurations.
- This patent can also be leveraged for machine part inspection, which necessitates high degrees of consistent accuracy
- An analog of probes in other machine and computer vision tasks is the notion of **keypoints**, which are used in descriptor-based feature matching algorithms such as SIFT (Scale-Invariant Feature Transform), SURF (Speeded-Up Robust Features), FAST (Features from Accelerated Segment Test), and ORB (Oriented FAST and rotated BRIEF). Many of these aforementioned approaches rely on computing gradients at specific, "interesting points" as is done here, and construct features for feature matching using a **Histogram of Oriented Gradients (HoG)** [1].

- For running this framework at multiple scales/resolutions, we want to use different probes at different scales.
- For multiscale, there is a need for running fast low-pass filtering. Can do so with rapid convolutions, which we will be discussing in a later patent in this course.
- Probes "contribute evidence" individually, and are not restricted to being on the pixel grid.
- The accuracy of this approach, similar to the other framework we looked at, is limited by the degree of quantization in the search space.

#### 1.2.3 Framework as Programming Objects

Let us also take a look at the object-oriented nature of this approach to better understand the framework we work with. One of these objects is the model:

**Model**: This has fields:

- **Probes**: This is the list of probe points. Note that we can also use **compiled probes**. In turn, each element in this list has fields:
  - **Position**
  - **Direction**
  - **Weight**
- **Granularity**: This is a scalar and is chosen during the training step.
- **Contrast**: This field is set to the computed contrast of the training/template pattern.

We can also look at some of the fields Generalized Degree of Freedom (**Generalized DOF**) object as well:

- **Rotation**
- **Shear** - The degree to which right angles are mapped into non-right angles.
- **Log size**
- **Log $x$ size**
- **Log $y$ size**

#### 1.2.4 Other Considerations for this Framework

We should also consider how to run our **translational search**. This search should be algorithmically conducted:

- Efficiently
- At different levels of resolution
- Hexagonally, rather than on a square grid - there is a $\frac{4}{\pi}$ advantage of work done vs. resolution. Here, hexagonal peak detection is used, and to break ties, we arbitrarily set 3 of the 6 inequalities as $\geq$, and the other 3 as $>$.

**What is pose?**
Pose is short for position and orientation, and is usually determined with respect to a reference coordinate system. In the patent's definition, it is the **"mapping from pattern to image coordinates that represents a specific transformation and superposition of a pattern onto an image."**

Next, let us look into addressing "noise", which can cause random matches to occur. Area under $S(\theta)$ curve captures the probability of random matches, and we can compensate by calculating error and subtracting it out of the results. However, even with this compensation, we are still faced with additional noise in the result.

Instead, we can try to assign scoring weights by taking the dot product between gradient vectors: $\hat{\mathbf{v}}_1 \cdot \hat{\mathbf{v}}_2 = \cos\theta$. But one disadvantage of this approach is that we end up quantizing pose space.

Finally, let us look at how we score the matches between template and runtime image configurations: **scoring functions**. Our options are:

- Normalized correlation (above)
- Simple peak finding
- Removal of random matches (this was our "N" factor introduced above)

## 1.3 References

1. Histogram of Oriented Gradients, https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients