## Vanishing points

One of the more interesting and familiar phenomena in perspective geometry is that parallel lines in the 3D world typically project to non-parallel lines in the image and in particular these projected lines intersect at a image point – called the *vanishing point*. Parallel lines are quite common in man made environments. The boundaries of floors, ceilings and doorways typically align with a natural XYZ orthogonal coordinate system and produce visible edges in images. There are typically many other lines in the scene that are parallel to each of these axes as well. For example, furniture such as desks and shelves has this sort of cube geometry and is often placed so that its coordinate system is parallel to the scene's coordinate system. This implies that there are groups of lines and edges in images which are the projections of parallel lines in the world and the lines in each group all point to a vanishing point.

In earlier lectures we have discussed how to detect edges in images. In this lecture we will examine the problem of finding vanishing points.

Let's begin by reviewing the basic model. Take a point $(X_0, Y_0, Z_0)$ in space and a direction $(T_X, T_Y, T_Z)$. This defines a line

$$(X_0, Y_0, Z_0) + t(T_X, T_Y, T_Z).$$

If the camera center $(X, Y, Z) = (0, 0, 0)$ does not lie on the line, then the camera center and the line together define a plane.

The image projection of the line is the intersection of this plane with the image projection plane, and this image line can be parameterized by $t$:

$$(x(t), y(t)) \;=\; (\; \frac{X_0 + T_X\, t}{Z_0 + T_Z\, t}, \frac{Y_0 + T_Y\, t}{Z_0 + T_Z\, t}) \; f$$

Note that it is not immediately obvious that $(x(t), y(t))$ define a line. The fact that it *does* define a line follows from the geometric argument above.

If $T_Z \neq 0$, then we can let $t \to \infty$ and we get

$$(x_v, y_v) \;=\; f(\; \frac{T_X}{T_Z}, \frac{T_Y}{T_Z}) \tag{1}$$

which is the *vanishing point* of the line. If $T_Z = 0$, then the 3D line lies in a constant $Z$ plane, and the image projection of the line is

$$(x(t), y(t)) \;=\; (\; \frac{X_0 + T_X\; t}{Z_0}, \frac{Y_0 + T_Y\; t}{Z_0}) \; f$$

As $t \to \infty$, we go to a point at infinity in direction $(T_x, T_y)$ in the image plane. Thus, if we have a set of parallel lines whose direction is perpendicular to the $Z$ axis, then the vanishing point is a point at infinity in the direction of the lines.

Notice that the vanishing point is only defined by the direction vector $(T_X, T_Y, T_Z)$, not by the point $(X_0, Y_0, Z_0)$. This means that we can vary the latter point however we like (not just along the line) and we will always get the same vanishing point. Thus, any set of parallel lines have the same vanishing point. Thus, *parallel lines in the world project to image lines that converge at a vanishing point.*

Also note the above derivations are similar to the analysis of translational camera motion which we saw in lecture 2. Why? When you translate the camera, all points in the scene travel along straight lines relative to the camera center. From the camera's perspective, there is no difference between translating the camera and keeping the world fixed versus translating the world and keeping the camera fixed. Recall that when you translate the camera, all points move away from the direction of heading. The direction of heading is thus mathematically equivalent to the vanishing point.

Finally, you may have heard of vanishing points in the context of classical painting and drawing. In particular, you may have heard of 1,2, and 3 point perspective. What do these refer to? In a scene where there are many 3D lines/edges that are parallel to the scene's $X, Y, Z$ axes, the image projection projection plane will typically contain up to three *finite* vanishing points. (It can contain more, for example, if there are additional sets of parallel lines.)

Suppose that the scene's $X, Z$ axes are north and west, and $Y$ is the gravity direction. If the camera is pointing in the scene's $X$ (or $Z$) direction and the camera's $y$ axis is parallel to gravity, then the image is a *one point perspective* since there is only one finite vanishing point, namely at the optical axis. The other two vanishing points are at infinity. A *two point perspective* arises, for example, when the camera's $Y$ axis is parallel to the line of gravity, but the camera's $X$ axis different from the scene's $X$ and $Z$ axes (and you can deduce that the camera's $Z$ axis is also different from the scene's $X$ and $Z$ axes). In this case, the scenes $X$ and $Z$ axes both produce finite vanishing points. But since the gravity direction $Y$ produces a vanishing point at infinity, there are only two finite vanishing points. Finally, if none of the three camera axes are parallel to the scene's $XYZ$ axes, then you have a *three point perspective*. There are three finite vanishing points.

[ASIDE: When looking at a two or three point perspective drawing, you should position your eye so that the axes subtend a 90 degree angle, since that would be the situation in the scene. Notice that this typically requires your eye is quite close to the paper/canvas.]

## Estimating a vanishing point

Suppose you have run a Canny edge detector (or some other edge detector) and so you have a set of image points and orientations $(x, y, \theta)$ where $\theta$ is the direction of the gradient of the image intensity, i.e. perpendicular to the edge. Each such triplet defines a line

$$(x - x_v, y - y_v) \cdot (\cos\theta, \sin\theta) = 0$$

where $(x_v, y_v)$ is the vanishing point.

Estimating the location of the vanishing point requires estimating the intersection of such lines. This problem would be trivial to solve except that: (1) only some subset of the lines in the 3D scene will be parallel to each other and so we don't know which image edges to use, and (2) the estimated values of $(x, y, \theta)$ typically are noisy.

Notice that this problem resembles the problem we saw last lecture in which we wanted to fit a line to a set of points. There, (1) we talked about inliers and outliers, namely points that belonged to a line and those that did not, and (2) the positions of the points that belonged to a line were noisy. I argued that when the percentage of outliers was non-negligable, a least squares fit on all the data didn't make much sense and suggested instead that we use a method such as the Hough transform or RANSAC. We will follow a similar line of thinking today for vanishing point estimation.

### "Hough transform" approach 1

We will consider two different approaches, both of which are of the Hough transform flavor since they involve voting for location $(x_v, y_v)$ of the vanishing point. For the first approach, let's first assume that the vanishing point lies within the limited field of view of the image. (This is a very strong assumption and in general we don't want to make it. But it is a good place to start.)

Given $(x_i, y_i, \theta_i)$, our model for a line through $(x_i, y_i)$ and perpendicular to $(\cos\theta, \sin\theta)$ is

$$(x - x_i, y - y_i) \cdot (\cos\theta, \sin\theta) = 0 \ .$$

Here is a sketch of an algorithm we could try (assuming $\theta \neq 0$):

```
for each edge element (xi, yi, thetai){
  for x in 1:Nx
    y = round(  (yi sin (theta) + (xi - x) cos(theta)) / sin( theta) )
  if (1 <=  y  <= Ny)
    count(x,y)++
}
find (x,y) that maximizes count(x,y)   // there may be several peaks
```

This algorithm will be very sensitive to noise[1], and so to make it work you would need to do more. For example, to account for noise in the $\theta_i$ estimate, you could add a loop over a small range of angles centered at $\theta_i$. You could also weight the vote by the inverse distance from $(x_i, y_i)$, since the errors in $\theta_i$ would lead to bigger errors in the location of the line as you move away from $(x_i, y_i)$. We will not bother further with these details, though, since we will introduce a better algorithm later this lecture.

Note the time complexity of this algorithm. If we have $n$ edges and an $N \times N$ image then we need $O(nN)$ operations. (Recall we are assuming the vanishing point lies within the image.)

### "Hough transform" approach 2

A second approach is to try to estimate the vanishing point directly by taking pairs of edges $(x_i, y_i, \theta_i)$ and $(x_j, y_j, \theta_j)$, computing the intersection of their lines, and then voting directly on the intersection points. Note that this assumes that $\theta_i \neq \theta_j$.

[1] and there are other problems, such as skipping y values when $|\tan\theta_i| > 1$

You can find lines using Gaussian elimination, but there is an equivalent (and slightly quicker) way to find the intersection of two lines. Let the two lines $l_i$ be

$$a_i x + b_i y + c_i = 0$$

where $i = 1, 2$. Think of 3D vectors $(a_i, b_i, c_i)$ and $(x, y, 1)$, where the latter lies on a projection plane $Z = 1$. Then vectors $(a_i, b_i, c_i)$ and $(x, y, 1)$ are orthogonal to each other. In particular, $(a_i, b_i, c_i)$ is orthogonal to the plane $\pi_i$ spanned by the origin and the line $l_i$.

We now have two planes $\pi_1$ and $\pi_2$, both of which pass through the origin. The intersection of these two planes must therefore be a line that passes throuh the origin. This line meets $Z = 1$ at precisely the intersection of $l_1$ and $l_2$. What is this line?

Since $(a_i, b_i, c_i)$ is orthogonal to plane $\pi_i$, it follows that the cross product vector

$$(wx, wy, w) = (a_1, b_1, c_1) \times (a_2, b_2, c_2)$$

must lie in *both planes* $\pi$, and hence it lies on the intersection of the two planes. Thus, to get the intersection of the two lines $l_1$ and $l_2$, we normalize the cross product vector $(x, y, z)$ so that it intersects $Z = 1$, and so the intersection point is $(\frac{x}{z}, \frac{y}{z})$

For example, suppose we have the two lines

$$3x + 4y + 2 = 0$$

$$2x - y = 0.$$

Then $(3, 4, 2) \times (2, -1, 0) = (2, 4, -11)$ and so the intersection point is $(-\frac{2}{11}, -\frac{4}{11})$.

Here is a similar algorithm as above, but which is based edge intersections.

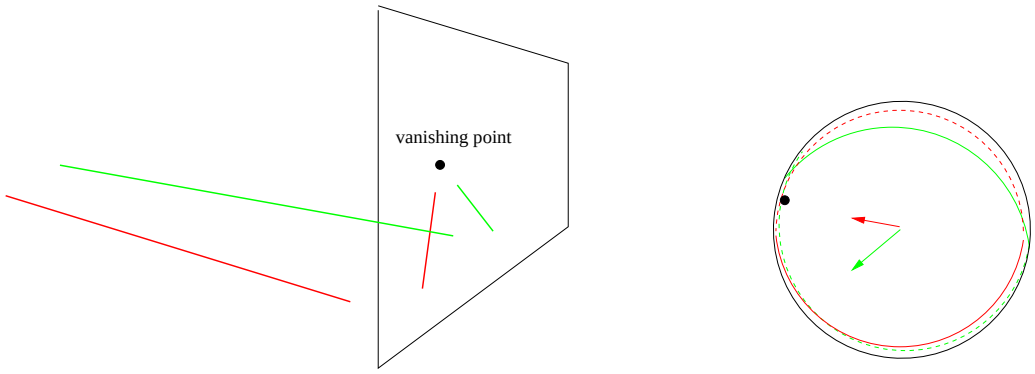
```
for each pair of edge elements (x,y,theta) and (x',y',theta')
  if theta != theta'{
    compute intersection (xv,yv) of lines containing these edges
    count( xv, yv )++
  }
find (xv,yv) that maximizes count(xv,yv)
```

The complexity is slightly different here. If there are $n$ edge elements, then there are $\frac{n(n-1)}{2}$ pairs, so the time is $O(n^2)$.

## Gaussian sphere

One key limitation with the above methods is that often the vanishing points does not lie within the image domain. The first algorithm explicitly assumed that the vanishing point was within the range of a fixed $N_x \times N_y$ grid. The second assumed a `count` matrix with real valued indices (since the `(xv,vy)` values are not necessarily integers. Obviously to implement the second algorithm you would need to quanitze the possible `(xv,vy)` values. However, no matter how bit a matrix you use for the possible quantized values of `(xv,vy)` values, it will still only cover a finite set of directions. The algorithm cannot handle the case that the vanishing point is at infinity, i.e. that the direction of the parallel lines have $T_Z = 0$ (see page 1).

The classic solution to this problem is to consider a projection sphere, rather than a projection plane. That is, place a unit sphere at the center of projection and consider each ray arriving at the camera center by where it intersects the unit sphere. This unit sphere is sometimes called the *Gaussian sphere*[2]




Recall that any edge element $(x, y, \theta)$ defines a line in the projection plane, and that this line defines a plane containing the line and the center of projection. This plane is sometimes called the *interpretation plane* in the vanishing point literature. Since this plane contains the center of projection, it must intersect the Gauss sphere on a *great circle*, namely a circle of radius 1 whose center is the center of projection. (Note that there is a 1-1 relationship between great circles and points on the unit sphere, namely each great circle lies on a plane which has a unit normal.) Thus, all points on the 3D line in question must project to points on the great circle defined by that line. In particular, the vanishing point must lies on that great circle as well.

A set of parallel lines in the scene define a set of great circles and since the vanishing point lies on each great circle, the vanishing point must be the intersection point of these great circles. There is nothing new here. I am just repeating the arguments made above for the projection plane, but now the projection plane (and the lines drawn on it) itself is projected onto the Gauss sphere.

The advantage of using the Gauss sphere rather than the projection plane is that the Gauss sphere is finite. We can re-define the two "Hough transform" methods above, but vote on the Gauss sphere instead. We just need to carve up the Gauss sphere into cells.[3] You would like to do so in such a way that the area of each tile is about constant. Why? Presumably the vanishing points could occur in any direction we will be looking for the tile with the maximum number of votes. If we were to make one tile much bigger than another, then we would be more likely to get the max number of votes in the big tile. If we don't use equal sized tile, then at least we should normalize the votes so that their number is weighted inversely by the tile area.

The first Hough method goes as follows. For each line element, you need to find the set of tiles on the unit sphere that intersect the interpretation plane defined by that line element. You increment the counter for each such tile. After all line elements have been considered, you pick the tile with the highest count. This is the vanishing point.

The second Hough method is to compute the intersection of pairs of lines as before. (This is

[2]The term "Gaussian sphere" is used for many things, not just this.

[3] There are various ways to tile a sphere. For example, you could use spherical coordinates (longitude and latitude). Or you could use a soccer ball type tiling.

equivalent to computing the intersection of the corresonding great circles.) You then normalize the intersection vector so it has unit length (not unit $Z$ value, which is what we did before). This defines a direction on the Gaussin sphere. Increment the counter for the tile containing this direction.

## Orthogonal frames (also known as "Manhattan world")

As mentioned at the beginning of the lecture, many scenes contain not just one set of parallel lines, but rather two or three sets of parallel lines and these lines directions are orthogonal. e.g. Many made environments for example have a floors and ceilings, walls and doorways, etc, which define a natural XYZ orthogonal coordinate system. (Such scenes are said to obey the *Manhattan World* model.)

On the one hand, having three vanishing points makes each of them more difficult to estimate, since the inlier edges for one vanishing point are outliers for the other vanishing points, i.e. we expect to find three maxima with our Hough method. On the other hand, the fact that the directions of the vanishing points are orthogonal is a constraint that could be useful both for finding the vanishing points and for subsequently using them e.g. to visually orient yourself with respect to the canonical directions in the scene.[4]

What can we say in general about the image positions of the vanishing points in the case? Suppose that there exist three vanishing points that correspond to three orthogonal directions. Let these vanishing points be $P, Q, R$ in the image projection plane $Z = f$. What can we say about the relationship between $P, Q, R$ ?

Since each of these points lies in the image projection plane $Z = f$, the line joining any two of these points to each other also lies in the projection plane. Let $C$ be the center of projection (the origin) and let $O$ be the principal point which is the intersection of the $Z$ axis with the projection plane. Then the vector $CO$ is perpendicular to the line between any two of the vanishing points.

Take a vanishing point $S$. Since vector $CS$ is perpendicular to vectors $CR$ and $CQ$, vector $CS$ is perpendicular to the vector $CR-CQ$. But vector $OC$ is also perpendicular to the vector $CR-CQ$. Thus $CR - CQ$ must be be perpendicular to the plane spanned by $OC$ and $CS$. Thus, within the projection plane $Z = f$, the line through $CR$ and $CQ$ must meet the line through $CS$ and $CO$ at right angles. Thus, $O$ lies on the unique image line from $S$ that meets line $RS$ at right angles.

Applying these arguments to the other two pairs, we get that the image center $O$ is at the intersection of the (three) lines through each of the vanishing points that meets at a right angle the line through the other two vanishing points. (This intersection point is called the *orthocenter* of the triangle.)

At first glance, it seems from the above argument that if we know the three vanishing points, then we can find the image center. Similarly, if we know two of the vanishing points and we know the image center, then we can find the third vanishing point.

Not so fast, however. The above arguments were in terms of points on the projection plane, rather than in terms of pixel units. But the image is given in pixel units! If we want to use the above result, then we need to know the relationship between the two, namely we need to know the calibration matrix $\mathbf{K}$. I will leave this issue for an Exercise.

[4]See for example, J. M. Coughlin and A.L. Yuille "Manhattan World: compass direction from a single image by Bayesian inference" in ICCV 1999", or more recently "J. P. Tardif "Non-iterative approach for fast and accurate vanishing point detection" in ICCV 2009, and several papers cited within.