2 Curves

2.1 Parametric Curves

There are multiple ways to represent curves in two dimensions:

- **Explicit:** $y = f(x)$, given x, find y.

  > *Example:*
  > The explicit form of a line is $y = mx + b$. There is a problem with this representation–what about vertical lines?

- **Implicit:** $f(x, y) = 0$, or in vector form, $f(\bar{p}) = 0$.

  > *Example:*
  > The implicit equation of a line through $\bar{p}_0$ and $\bar{p}_1$ is
  >
  > $$(x - x_0)(y_1 - y_0) - (y - y_0)(x_1 - x_0) = 0.$$
  >
  > Intuition:
  > - The direction of the line is the vector $\vec{d} = \bar{p}_1 - \bar{p}_0$.
  > - So a vector from $\bar{p}_0$ to any point on the line must be parallel to $\vec{d}$.
  > - Equivalently, any point on the line must have direction from $\bar{p}_0$ perpendicular to $\vec{d}^{\perp} = (d_y, -d_x) \equiv \vec{n}$.
  >   This can be checked with $\vec{d} \cdot \vec{d}^{\perp} = (d_x, d_y) \cdot (d_y, -d_x) = 0$.
  > - The vector $\vec{n} = (y_1 - y_0, x_0 - x_1)$ is called a **normal** vector.
  > - So, putting it all together, for any point $\bar{p}$ on the line, we know that $(\bar{p} - \bar{p}_0) \cdot \vec{n} = 0$. This is the general form for a line through $\bar{p}_0$ that is perpendicular (normal) to $\vec{n}$. If you then substitute $\vec{n} = (y_1 - y_0, x_0 - x_1)$ into this implicit equation and simplify the dot product, you get the implicit equation above.

  > *Example:*
  > The implicit equation for a circle of radius r and center $\bar{p}_c = (x_c, y_c)$ is
  >
  > $$(x - x_c)^2 + (y - y_c)^2 = r^2,$$
  >
  > or in vector form,
  > $$\|\bar{p} - \bar{p}_c\|^2 = r^2.$$

- **Parametric:** $\bar{p} = \bar{f}(\lambda)$ where $\bar{f} : \mathbb{R} \rightarrow \mathbb{R}^2$, may be written as $\bar{p}(\lambda)$ or $(x(\lambda), y(\lambda))$.

Example:

A parametric line through $\bar{p}_0$ and $\bar{p}_1$ is

$$\bar{p}(\lambda) = \bar{p}_0 + \lambda \vec{d},$$

where $\vec{d} = \bar{p}_1 - \bar{p}_0$.

Note that bounds on λ must be specified:

- Line segment from $\bar{p}_0$ to $\bar{p}_1$: $0 \leq \lambda \leq 1$.
- Ray from $\bar{p}_0$ in the direction of $\bar{p}_1$: $0 \leq \lambda < \infty$.
- Line passing through $\bar{p}_0$ and $\bar{p}_1$: $-\infty < \lambda < \infty$

Example:

What's the perpendicular bisector of the line segment between $\bar{p}_0$ and $\bar{p}_1$?

- The midpoint is $\bar{p}(\lambda)$ where $\lambda = \frac{1}{2}$, that is, $\bar{p}_0 + \frac{1}{2}\vec{d} = \frac{\bar{p}_0 + \bar{p}_1}{2}$.
- The line perpendicular to $\bar{p}(\lambda)$ has direction parallel to the normal of $\bar{p}(\lambda)$, which is $\vec{n} = (y_1 - y_0, -(x_1 - x_0))$.

Hence, the perpendicular bisector is the line $\ell(\alpha) = \left(\bar{p}_0 + \frac{1}{2}\vec{d}\right) + \alpha \vec{n}$.

Example:

Find the intersection of the lines $\bar{l}(\lambda) = \bar{p}_0 + \lambda \vec{d_0}$ and $f(\bar{p}) = (\bar{p} - \bar{p}_1) \cdot \vec{n}_1 = 0$.

Substitute $\bar{l}(\lambda)$ into the implicit equation $f(\bar{p})$ to see what value of λ satisfies it:

$$
\begin{aligned}
f\left(\bar{l}(\lambda)\right) &= \left(\bar{p}_0 + \lambda \vec{d_0} - \bar{p}_1\right) \cdot \vec{n}_1 \\
&= \lambda \vec{d_0} \cdot \vec{n}_1 - (\bar{p}_1 - \bar{p}_0) \cdot \vec{n}_1 \\
&= 0
\end{aligned}
$$

Therefore, if $\vec{d_0} \cdot \vec{n}_1 \neq 0$,

$$\lambda^* = \frac{(\bar{p}_1 - \bar{p}_0) \cdot \vec{n}_1}{\vec{d_0} \cdot \vec{n}_1},$$

and the intersection point is $\bar{l}(\lambda^*)$. If $\vec{d_0} \cdot \vec{n}_1 = 0$, then the two lines are parallel with no intersection or they are the same line.

Example:
The parametric form of a circle with radius r for $0 \le \lambda < 1$ is

$$\bar{p}(\lambda) = (r\cos(2\pi\lambda), r\sin(2\pi\lambda)).$$

This is the polar coordinate representation of a circle. There are an infinite number of parametric representations of most curves, such as circles. Can you think of others?

An important property of parametric curves is that it is easy to generate points along a curve by evaluating $\bar{p}(\lambda)$ at a sequence of λ values.

2.1.1   Tangents and Normals

The **tangent** to a curve at a point is the instantaneous direction of the curve. The line containing the tangent intersects the curve at a point. It is given by the derivative of the parametric form $\bar{p}(\lambda)$ with regard to λ. That is,

$$\vec{\tau}(\lambda) = \frac{d\bar{p}(\lambda)}{d\lambda} = \left(\frac{dx(\lambda)}{d\lambda}, \frac{dy(\lambda)}{d\lambda} \right).$$

The **normal** is perpendicular to the tangent direction. Often we normalize the normal to have unit length. For closed curves we often talk about an inward-facing and an outward-facing normal. When the type is unspecified, we are usually dealing with an outward-facing normal.
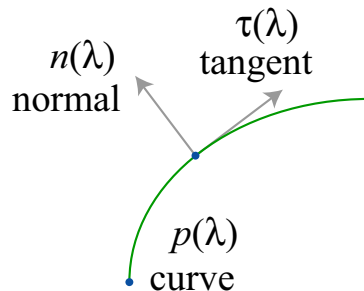
We can also derive the normal from the implicit form. The normal at a point $\bar{p} = (x, y)$ on a curve defined by $f(\bar{p}) = f(x, y) = 0$ is:

$$\vec{n}(\bar{p}) = \nabla f(\bar{p})|_{\bar{p}} = \left(\frac{\partial f(x, y)}{\partial x}, \frac{\partial f(x, y)}{\partial y} \right)$$

Derivation:
For any curve in implicit form, there also exists a parametric representation $\bar{p}(\lambda) =$

$(x(\lambda), y(\lambda))$. All points on the curve must satisfy $f(\bar{p}) = 0$. Therefore, for any choice of λ, we have:

$$0 = f(x(\lambda), y(\lambda))$$

We can differentiate both side with respect to λ:

$$0 = \frac{d}{d\lambda} f(x(\lambda), y(\lambda)) \tag{1}$$

$$0 = \frac{\partial f}{\partial x} \frac{dx(\lambda)}{d\lambda} + \frac{\partial f}{\partial y} \frac{dy(\lambda)}{d\lambda} \tag{2}$$

$$0 = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right) \cdot \left(\frac{dx(\lambda)}{d\lambda}, \frac{dy(\lambda)}{d\lambda} \right) \tag{3}$$

$$0 = \nabla f(\bar{p})|_{\bar{p}} \cdot \vec{\tau}(\lambda) \tag{4}$$

This last line states that the gradient is perpendicular to the curve tangent, which is the definition of the normal vector.


Example:
The implicit form of a circle at the origin is: $f(x, y) = x^2 + y^2 - R^2 = 0$. The normal at a point (x, y) on the circle is: $\nabla f = (2x, 2y)$.

Exercise: show that the normal computed for a line is the same, regardless of whether it is computed using the parametric or implicit forms. Try it for another surface.

2.2 Ellipses

- **Implicit:** $\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$. This is only for the special case where the ellipse is centered at the origin with the major and minor axes aligned with $y = 0$ and $x = 0$.
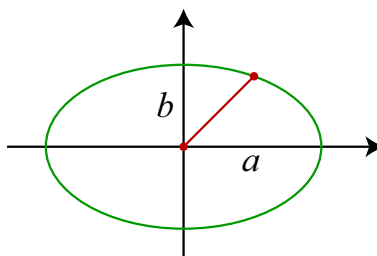
- **Parametric:** $x(\lambda) = a \cos(2\pi\lambda), \; y(\lambda) = b \sin(2\pi\lambda)$, or in vector form

$$\bar{p}(\lambda) = \left[\begin{array}{c} a \cos(2\pi\lambda) \\ b \sin(2\pi\lambda) \end{array} \right].$$

The implicit form of ellipses and circles is common because there is no explicit functional form. This is because y is a multifunction of x.

2.3 Polygons

A **polygon** is a continuous, piecewise linear, closed planar curve.

- A **simple** polygon is non self-intersecting.

- A **regular** polygon is simple, equilateral, and equiangular.

- An **n-gon** is a regular polygon with n sides.

- A polygon is **convex** if, for any two points selected inside the polygon, the line segment between them is completely contained within the polygon.

Example:
To find the vertices of an n-gon, find n equally spaced points on a circle.
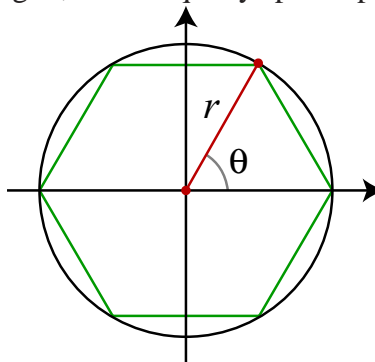
In polar coordinates, each vertex $(x_i, y_i) = (r\cos(\theta_i), r\sin(\theta_i))$, where $\theta_i = i\frac{2\pi}{n}$ for $i = 0 \ldots n-1$.

- To translate: Add (x_c, y_c) to each point.
- To scale: Change r.
- To rotate: Add $\Delta\theta$ to each θ_i.

2.4 Rendering Curves in OpenGL

OpenGL does not directly support rendering any curves other that lines and polylines. However, you can sample a curve and draw it as a line strip, e.g.,:

```
float x, y;
glBegin(GL_LINE_STRIP);
for (float t=0 ; t <= 1 ; t += .01)
```

```
    computeCurve( t, &x, &y);
    glVertex2f(x, y);
}
glEnd();
```

You can adjust the step-size to determine how many line segments to draw. Adding line segments will increase the accuracy of the curve, but slow down the rendering.

The GLU does have some specialized libraries to assist with generating and rendering curves. For example, the following code renders a disk with a hole in its center, centered about the z-axis.

```
GLUquadric q = gluNewQuadric();
gluDisk(q, innerRadius, outerRadius, sliceCount, 1);
gluDeleteQuadric(q);
```

See the OpenGL Reference Manual for more information on these routines.