Ray Tracing, Part IV

Geoffrey Matthews

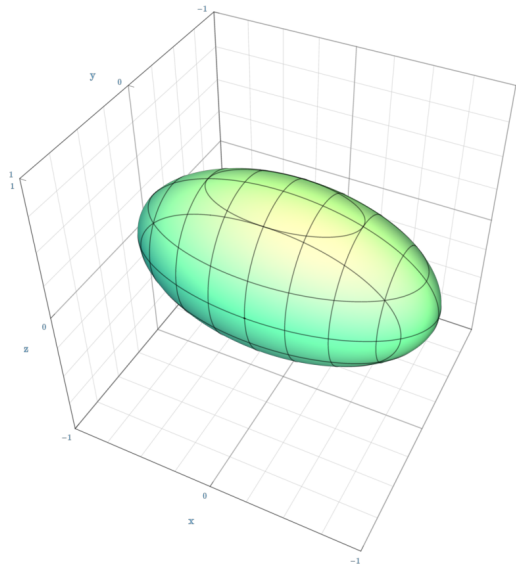
Department of Computer Science
Western Washington University

Fall 2015

Readings

- https://en.wikipedia.org/wiki/Gradient
- https://en.wikipedia.org/wiki/Quadric
- http://www.win.tue.nl/~sterk/Bouwkunde/hoofdstuk3.pdf

Ellipsoids



$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$$

- How can we intersect a ray with this surface?
- How can we find the normal?
- How can we rotate this shape?

Intersect a ray with an ellipsoid

Constraint on the surface:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$$

Points on the line generated by:

$$p + tv = (p_0, p_1, p_2) + t(v_0, v_1, v_2)$$

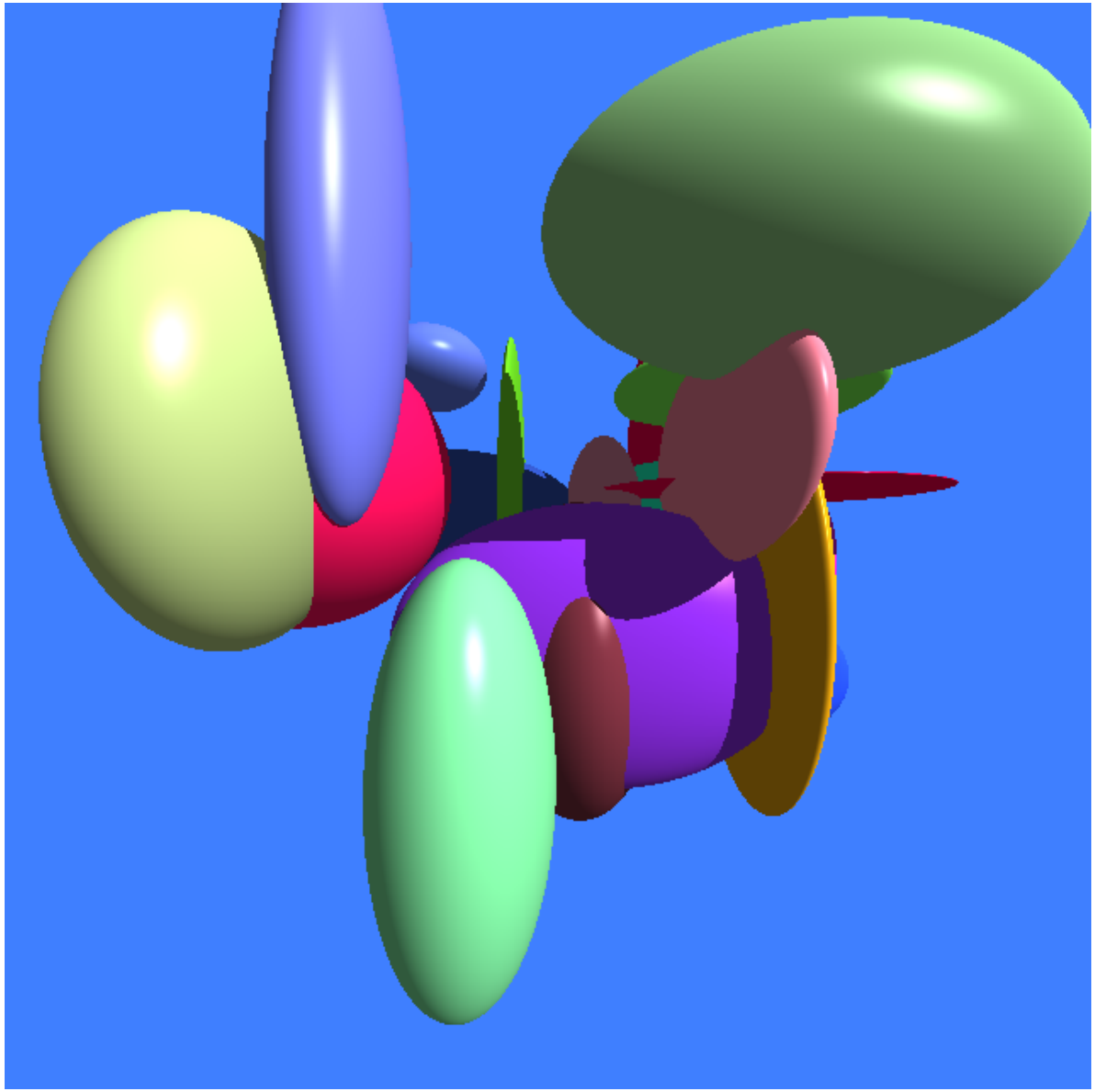
Plug one into the other to satisfy both:

$$\frac{(p_0 + tv_0)^2}{a^2} + \frac{(p_1 + tv_1)^2}{b^2} + \frac{(p_2 + tv_2)^2}{c^2} = 1$$

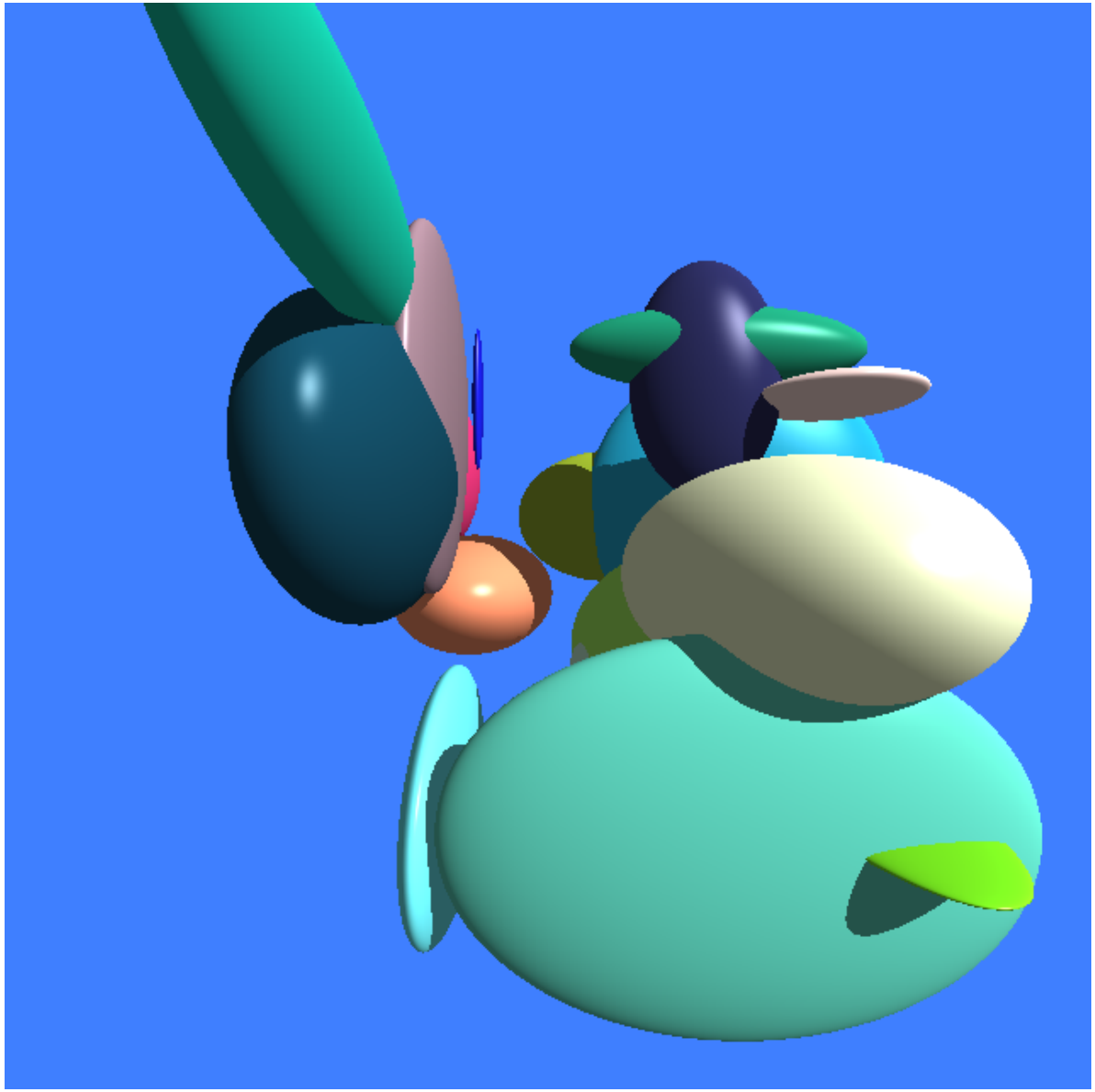
Collect terms to get our quadratic to solve:

$$\left(\frac{v_0^2}{a^2} + \frac{v_1^2}{b^2} + \frac{v_2^2}{c^2}\right) t^2 + \left(\frac{2p_0v_0}{a^2} + \frac{2p_1v_1}{b^2} + \frac{2p_2v_2}{c^2}\right) t + \left(\frac{p_0^2}{a^2} + \frac{p_1^2}{b^2} + \frac{p_2^2}{c^2}\right)$$

How do we move ellipsoids away from the origin?



What about the normal for an ellipsoid?



Calculus to the rescue!

- Ellipsoids satisfy the constraint:

$$f(x, y, z) = \frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$$

- This means that the function f is *constant* across the surface. If we take the *gradient* of f, it will be a vector pointing in the direction of maximum change of f, which will be perpendicular to the directions in which it is not changing at all.
- The gradient is defined as

$$\nabla f(x, y, z) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \right)$$

- The partial derivative of a function, for example, $\frac{\partial f}{\partial x}$, is simply the derivative of f treating everything except x as a constant.

Normals for the ellipsoid

- For the ellipsoid, the function is

$$f(x, y, z) = \frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2}$$

- What is the gradient of this function?

Normals for the ellipsoid

- For the ellipsoid, the function is

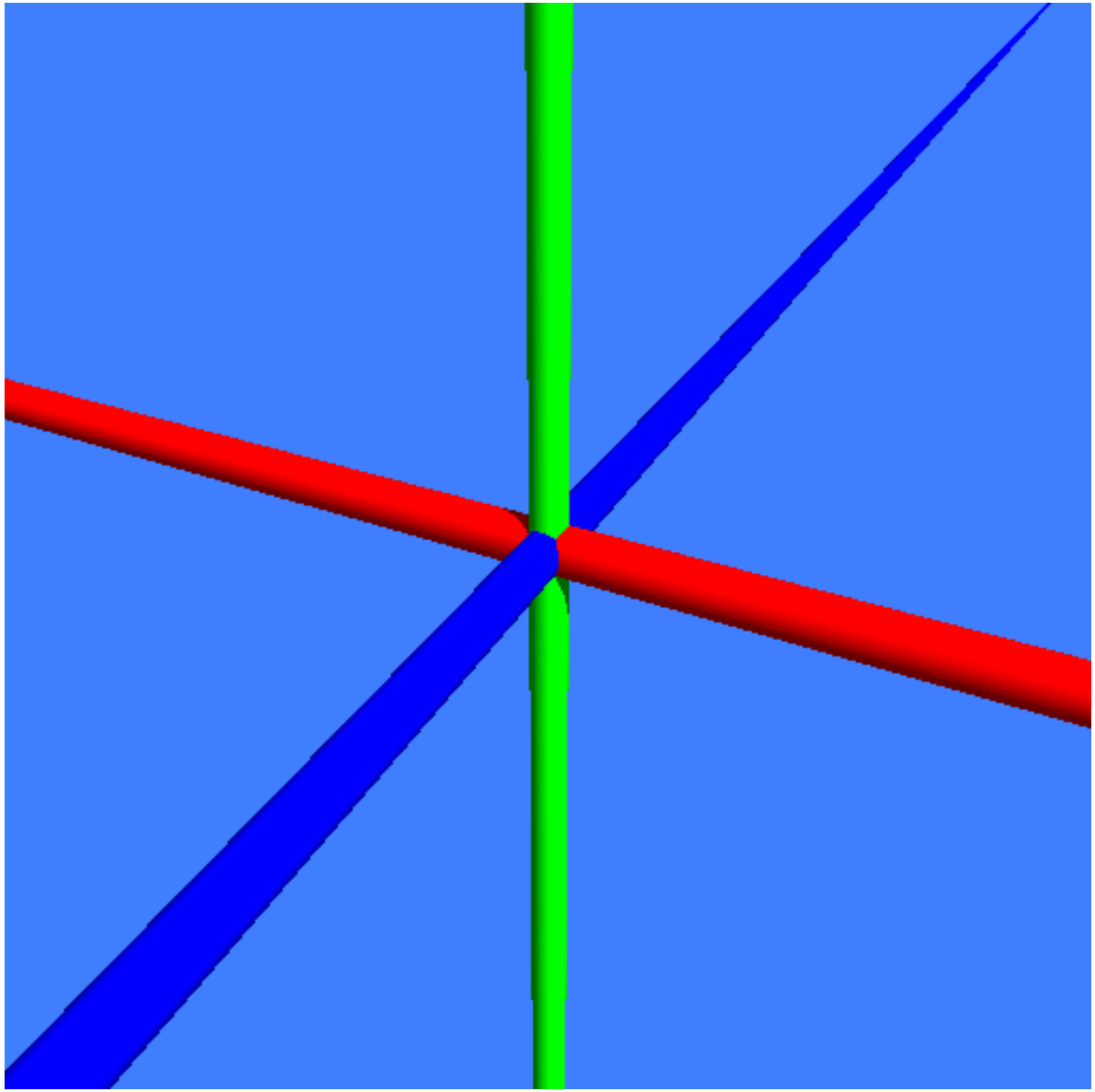
$$f(x, y, z) = \frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2}$$

- What is the gradient of this function?

$$\nabla f(x, y, z) = \left(\frac{2x}{a^2}, \frac{2y}{b^2}, \frac{2z}{c^2} \right)$$

- Since we want to normalize this anyway, we can drop the factor of 2.
- Note that this is in coordinates where the ellipsoid has not been translated. After we find our translated intersection point, we have to translate it back by subtracting the ellipsoid's center before calculating the normal.
- Or just use the translated ray to find the (x, y, z) for the gradient while calculating the intersection.

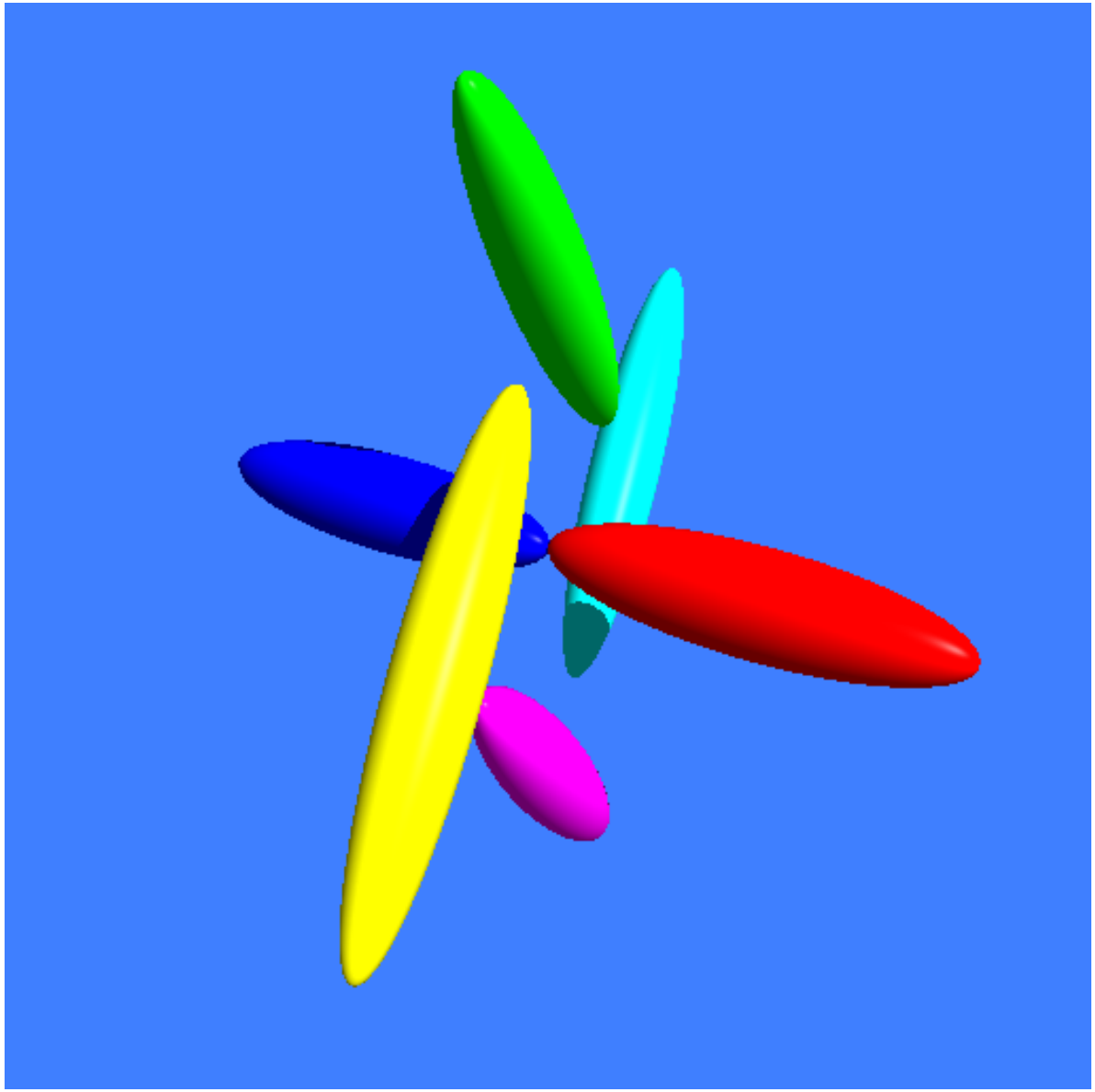
Can use ellipsoids for lots of things



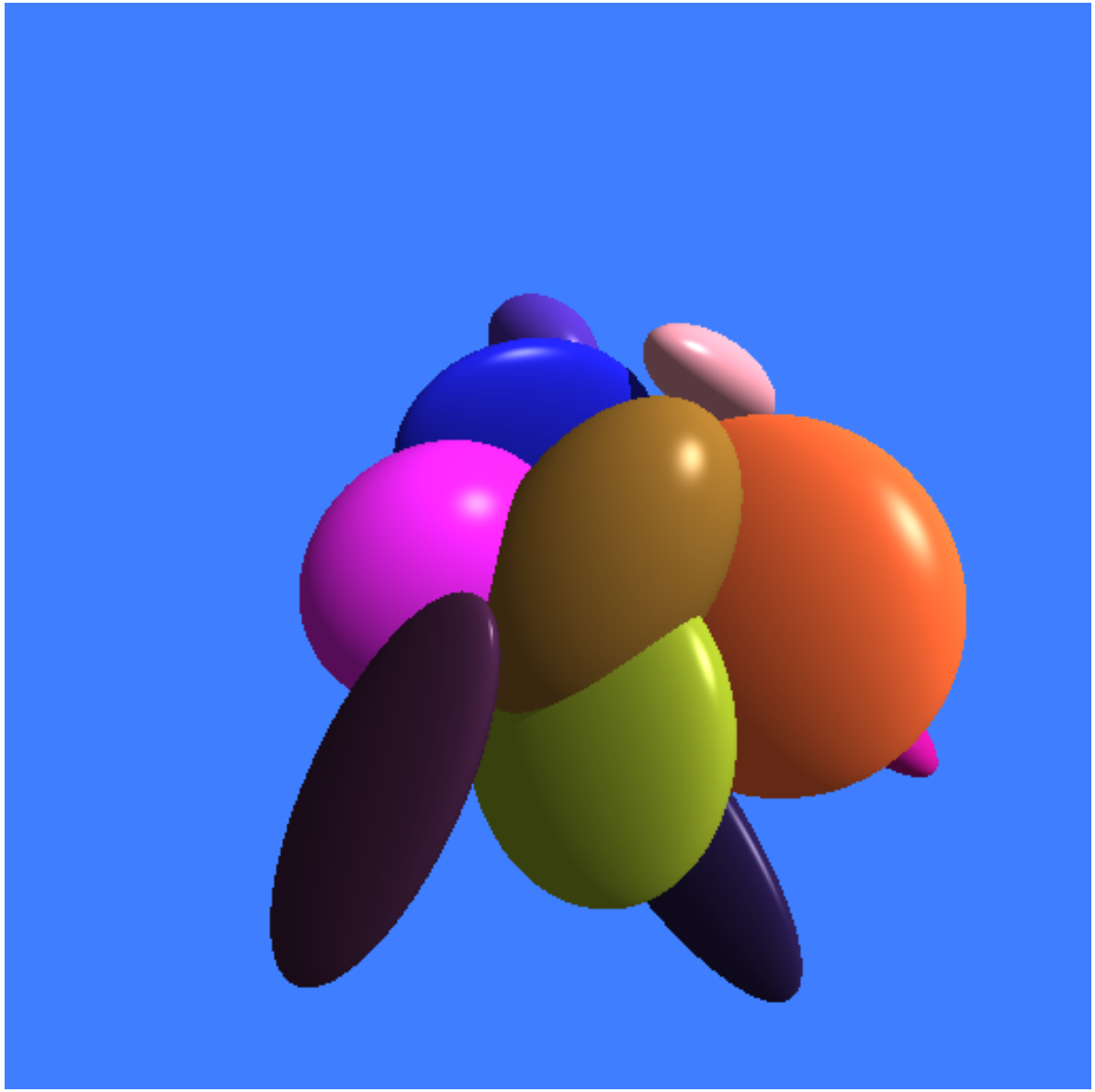
Other quadrics

- Using the tips above for the ellipsoid, you should be able to render any of the quadrics from this page: `https://en.wikipedia.org/wiki/Quadric`

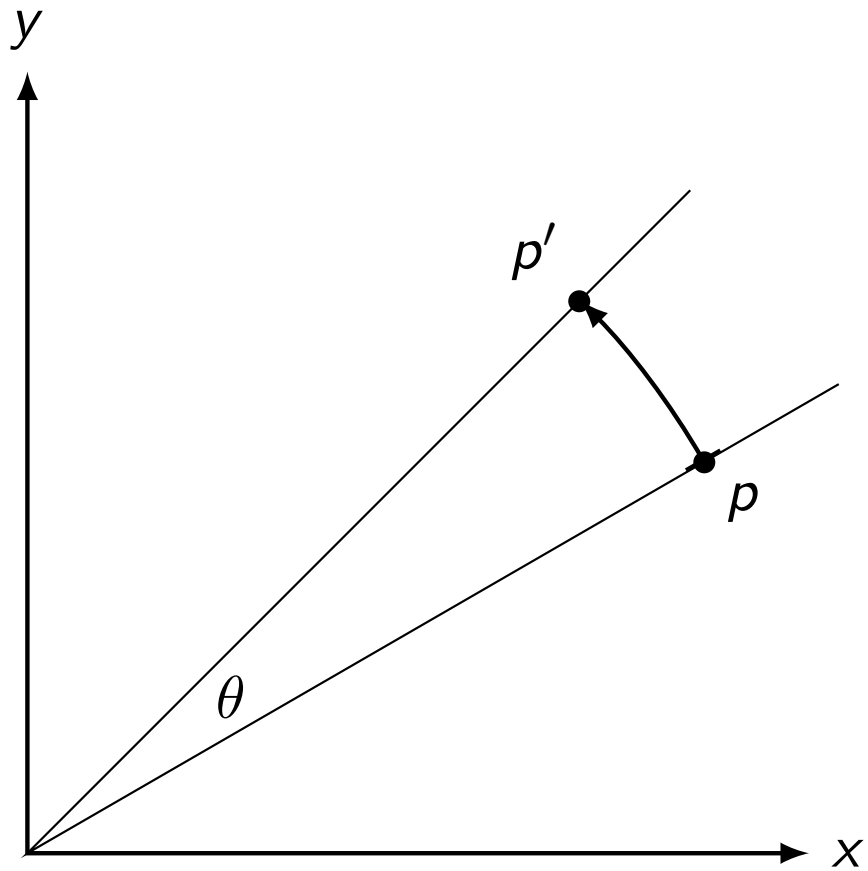
Rotating ellipsoids



How do we rotate quadrics?

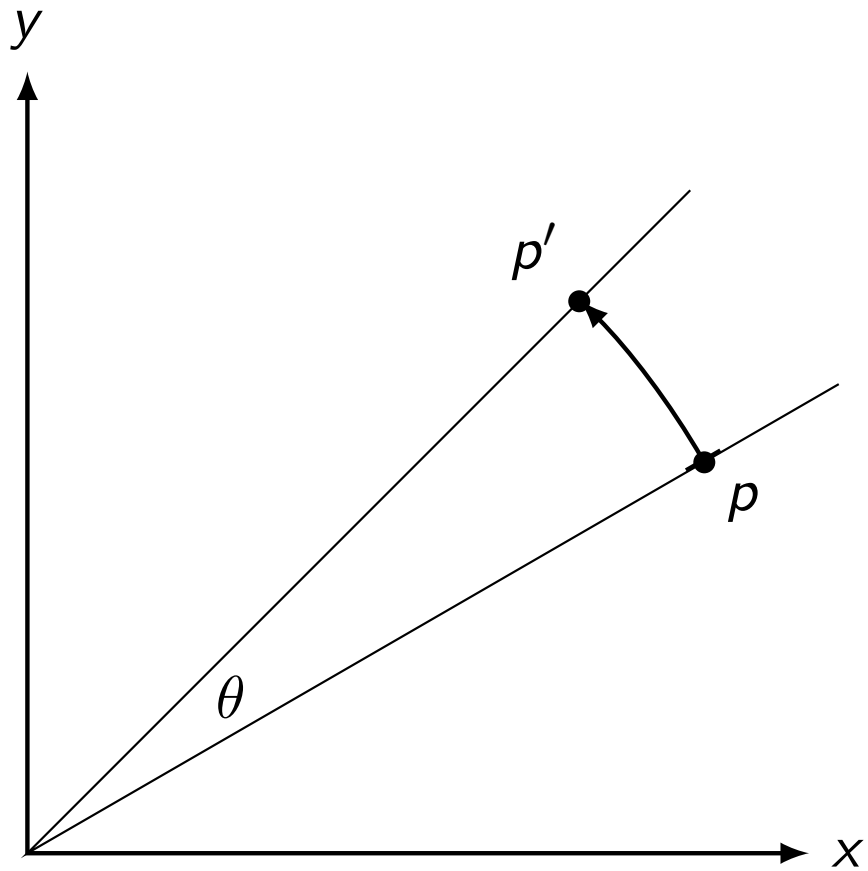


Rotating points



$$
\begin{aligned}
x' &= x\cos(\theta) - y\sin(\theta) \\
y' &= x\sin(\theta) + y\cos(\theta)
\end{aligned}
$$

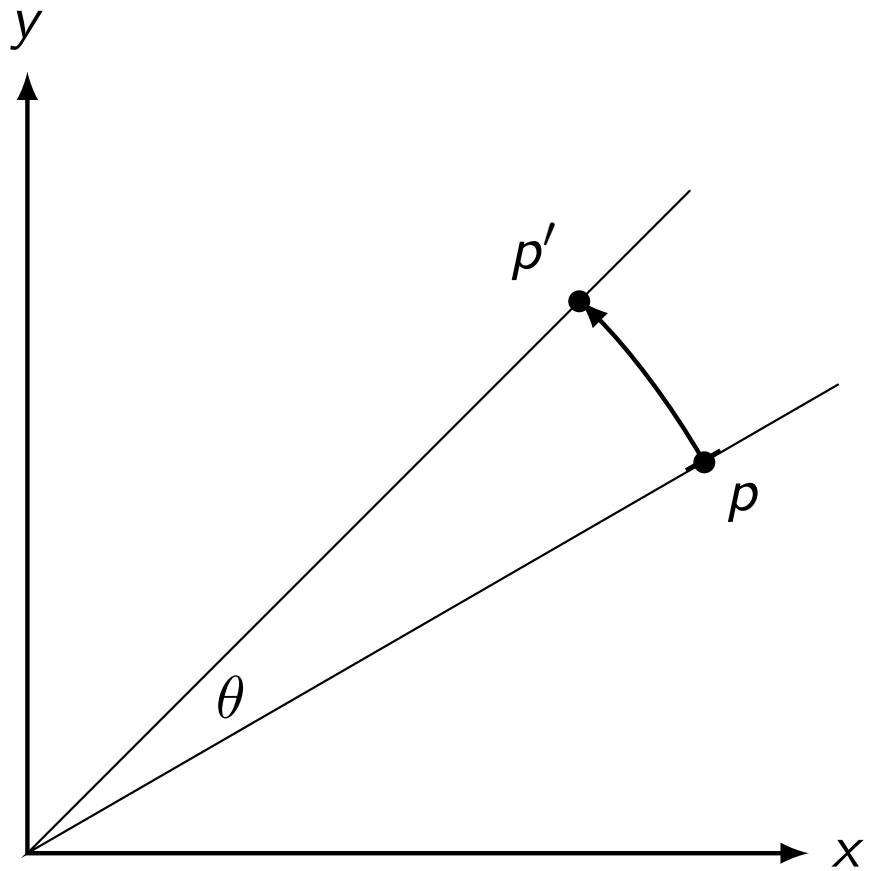
Rotating points



$$\begin{aligned} x' &= x\cos(\theta) - y\sin(\theta) \\ y' &= x\sin(\theta) + y\cos(\theta) \end{aligned}$$

- What if we want to rotate in the opposite direction?

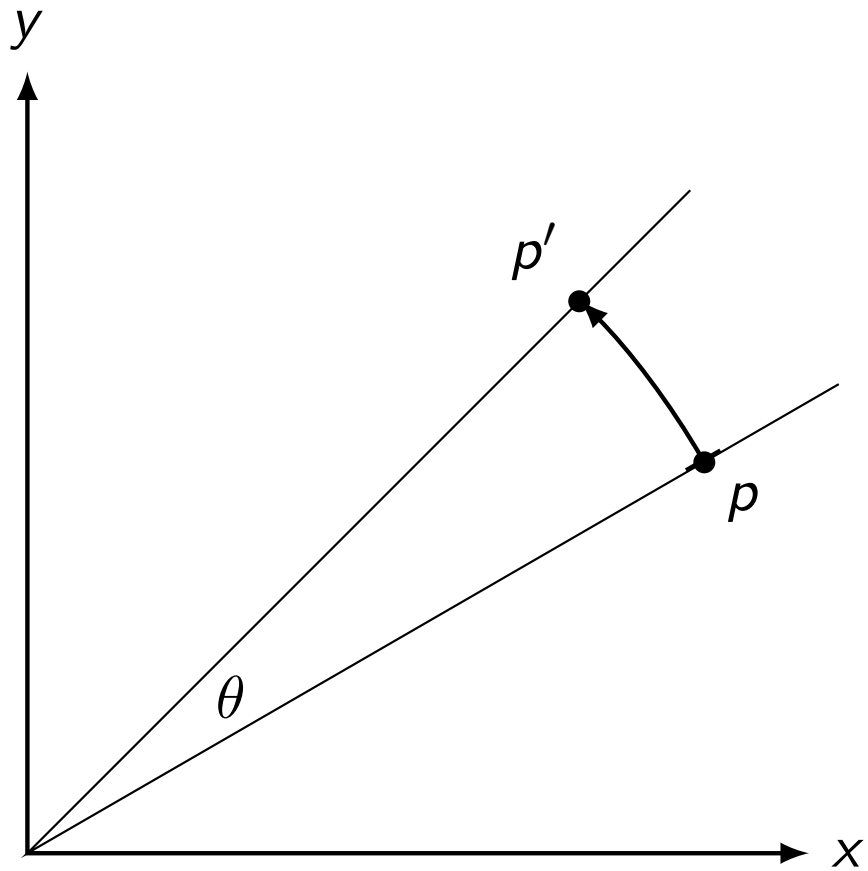
Rotating points



$$\begin{aligned} x' &= x\cos(\theta) - y\sin(\theta) \\ y' &= x\sin(\theta) + y\cos(\theta) \end{aligned}$$

- What if we want to rotate in the opposite direction?
- If z is pointing in, is this a
  - right-handed frame?
  - right-handed rotation?

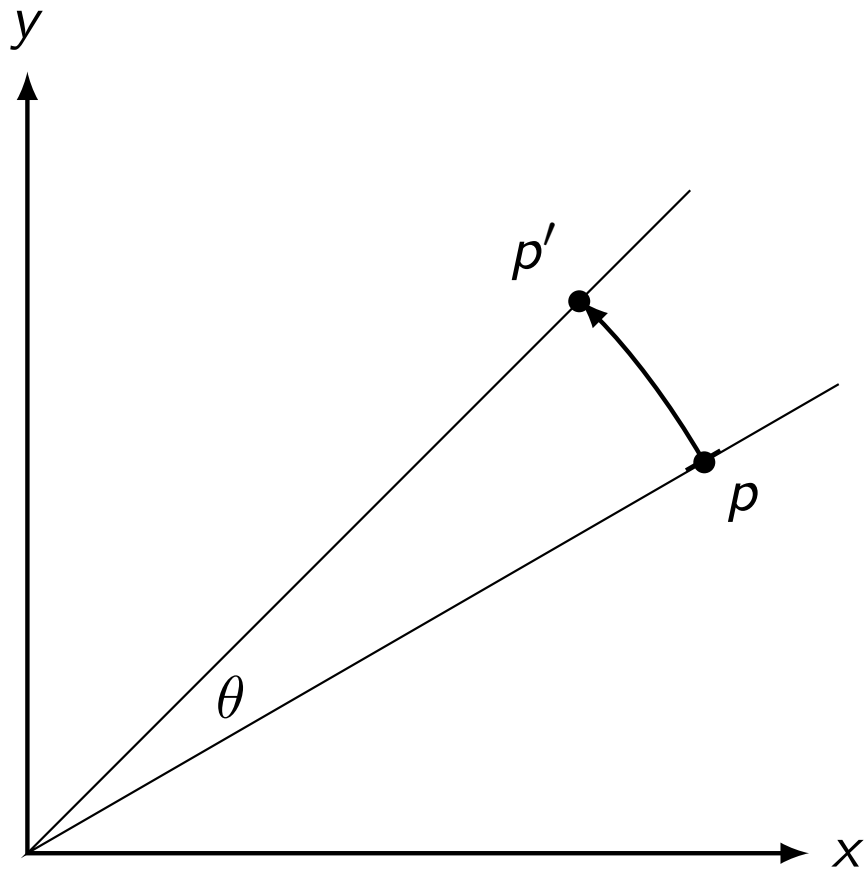
Rotating points



$$\begin{aligned} x' &= x\cos(\theta) - y\sin(\theta) \\ y' &= x\sin(\theta) + y\cos(\theta) \end{aligned}$$

- What if we want to rotate in the opposite direction?
- If z is pointing in, is this a
  - right-handed frame?
  - right-handed rotation?
- What about z pointing out?

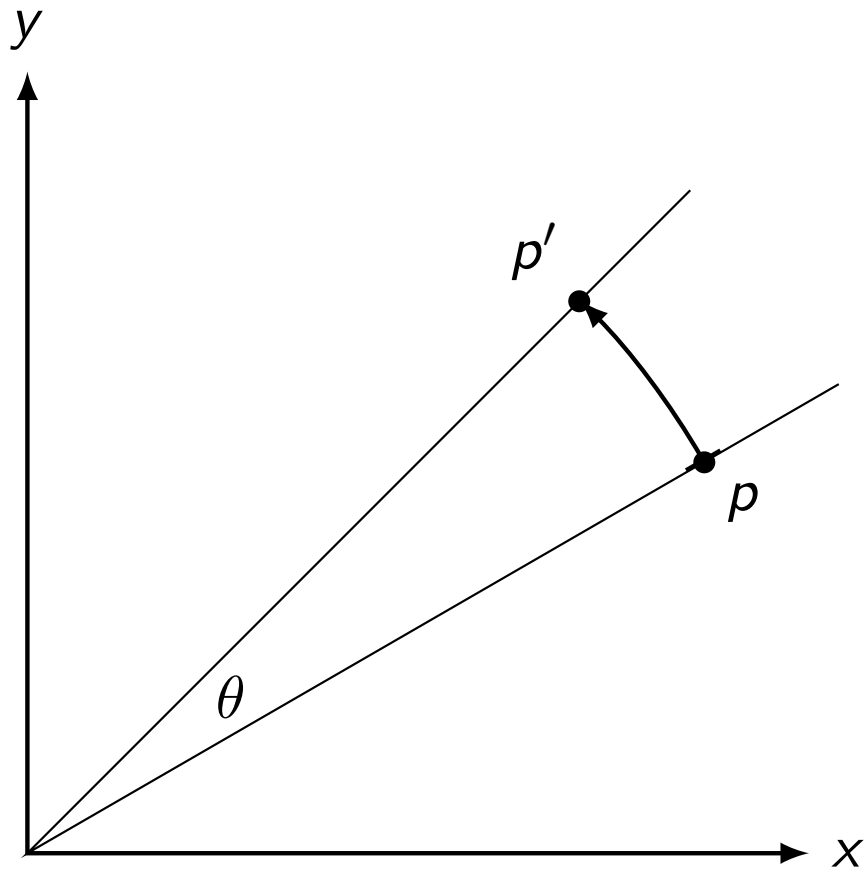
Rotating points



$$\begin{aligned} x' &= x\cos(\theta) - y\sin(\theta) \\ y' &= x\sin(\theta) + y\cos(\theta) \end{aligned}$$

- What if we want to rotate in the opposite direction?
- If z is pointing in, is this a
  - right-handed frame?
  - right-handed rotation?
- What about z pointing out?
- What about rotating vectors?

Rotating points



$$\begin{aligned} x' &= x\cos(\theta) - y\sin(\theta) \\ y' &= x\sin(\theta) + y\cos(\theta) \end{aligned}$$

- What if we want to rotate in the opposite direction?
- If z is pointing in, is this a
  - right-handed frame?
  - right-handed rotation?
- What about z pointing out?
- What about rotating vectors?

For derivation of equation:
`https://engineering.purdue.edu/~bethel/rot2.pdf`

Rotating in 3D

- Rotating around the z axis:

$$p' = (x\cos(\theta) - y\sin(\theta), x\sin(\theta) + y\cos(\theta), z)$$

- Rotating around the y axis:

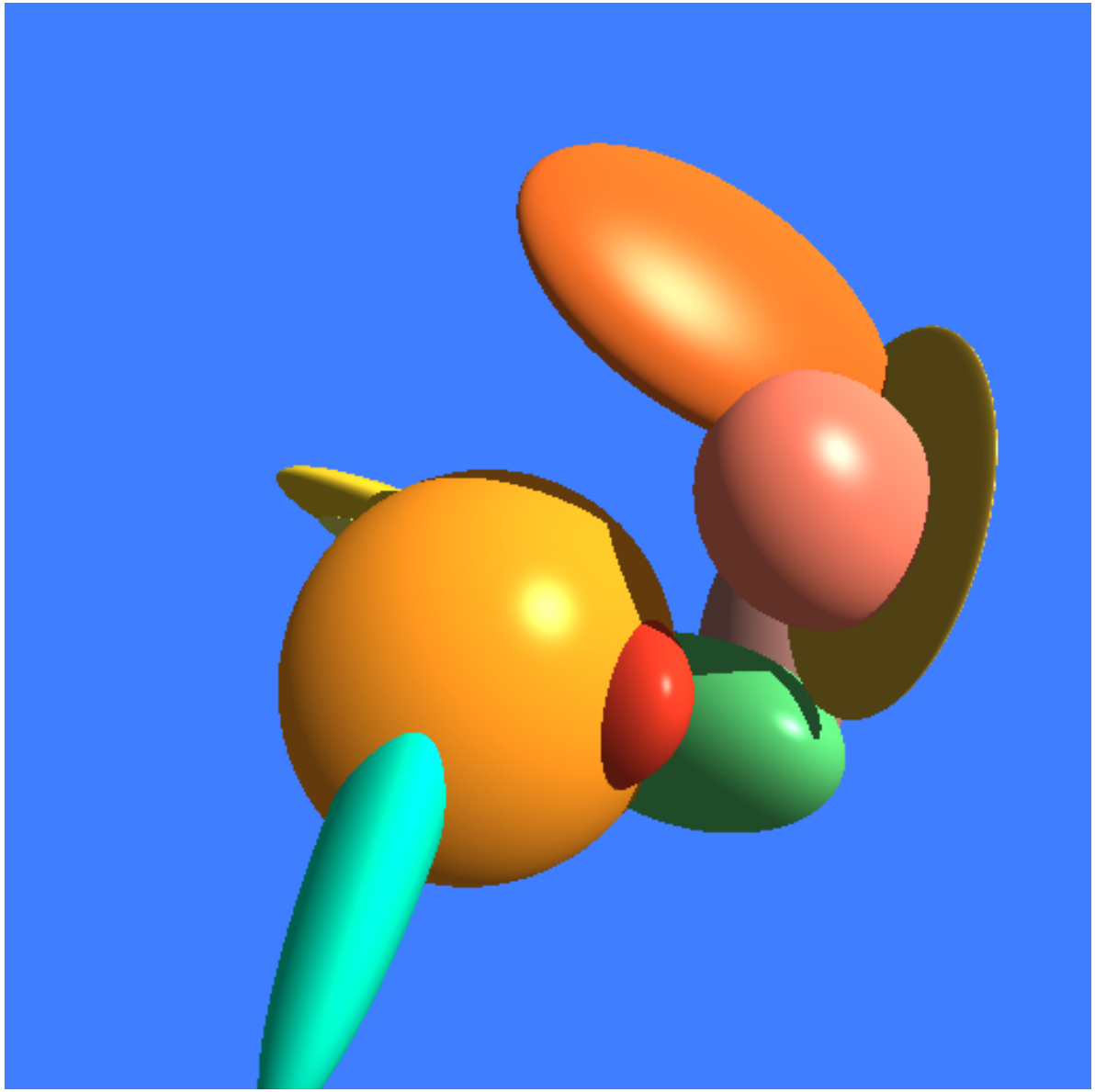
$$p' = (x\cos(\theta) - z\sin(\theta), y, x\sin(\theta) + z\cos(\theta))$$

- Rotating around the x axis:

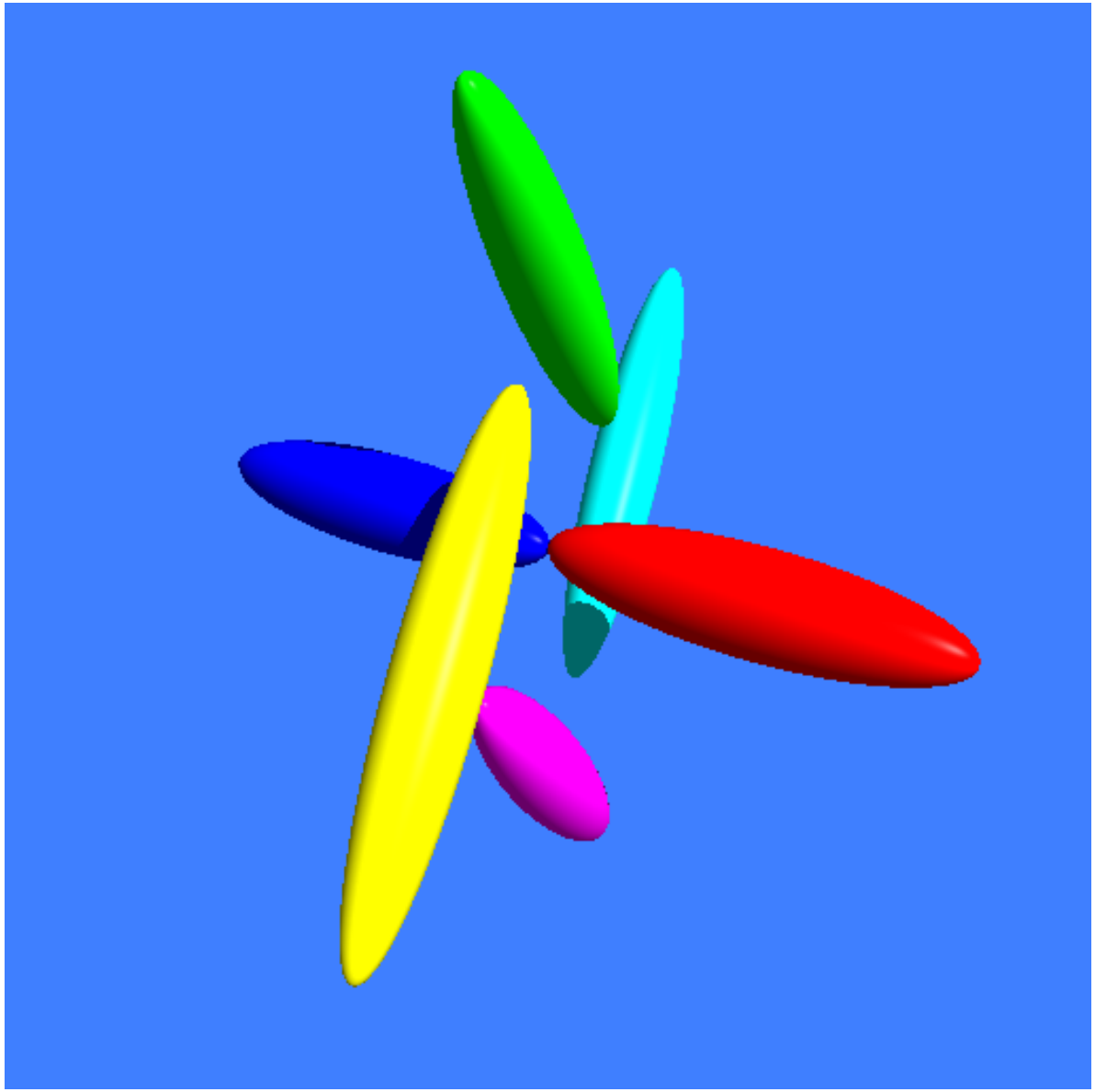
$$p' = (x, y\cos(\theta) - z\sin(\theta), y\sin(\theta) + z\cos(\theta))$$

- General rotations can be built up as sequences of x, y and z rotations.

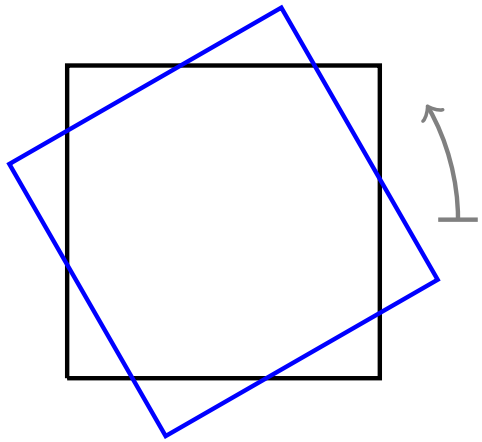
Rotating quadrics



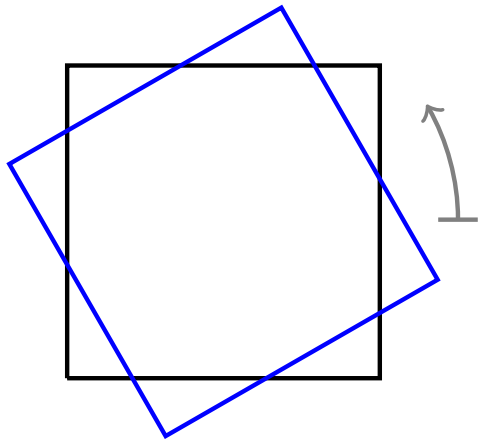
Rotating quadrics

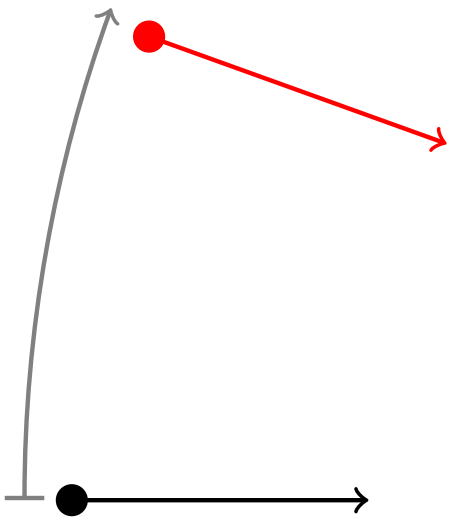


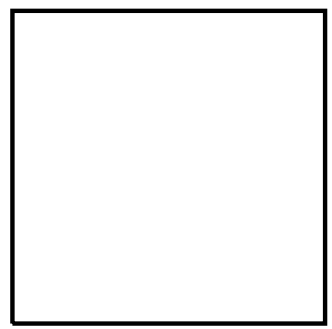
What about rotating quadrics?



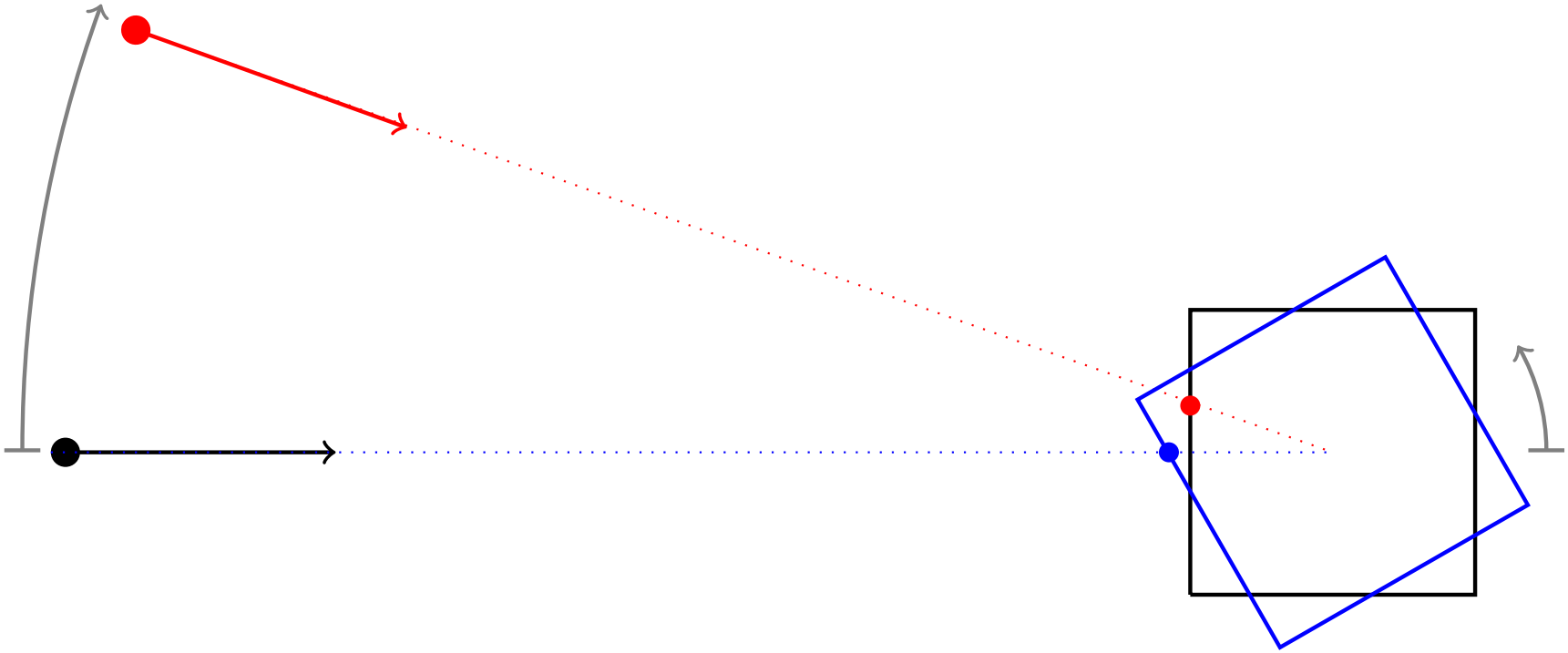
What about rotating quadrics?



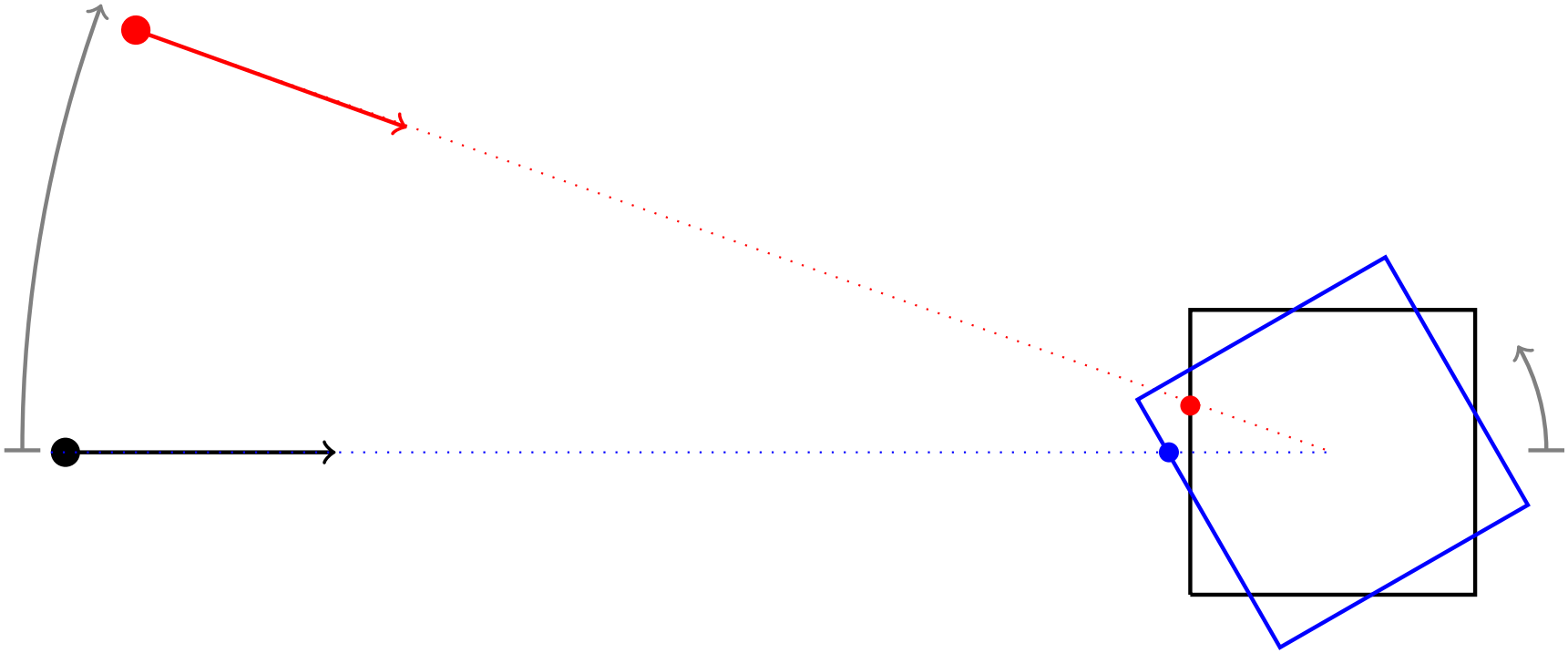




How do we find the intersection point?

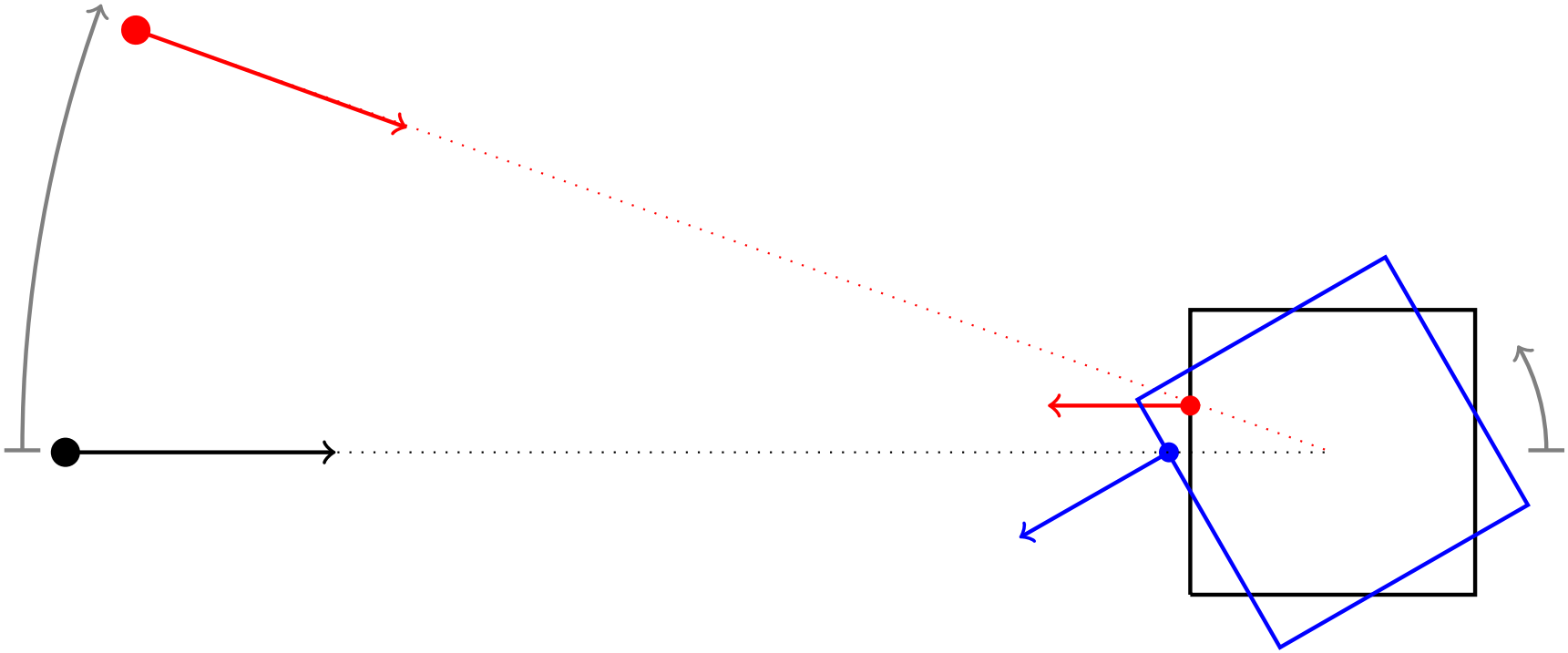


How do we find the intersection point?

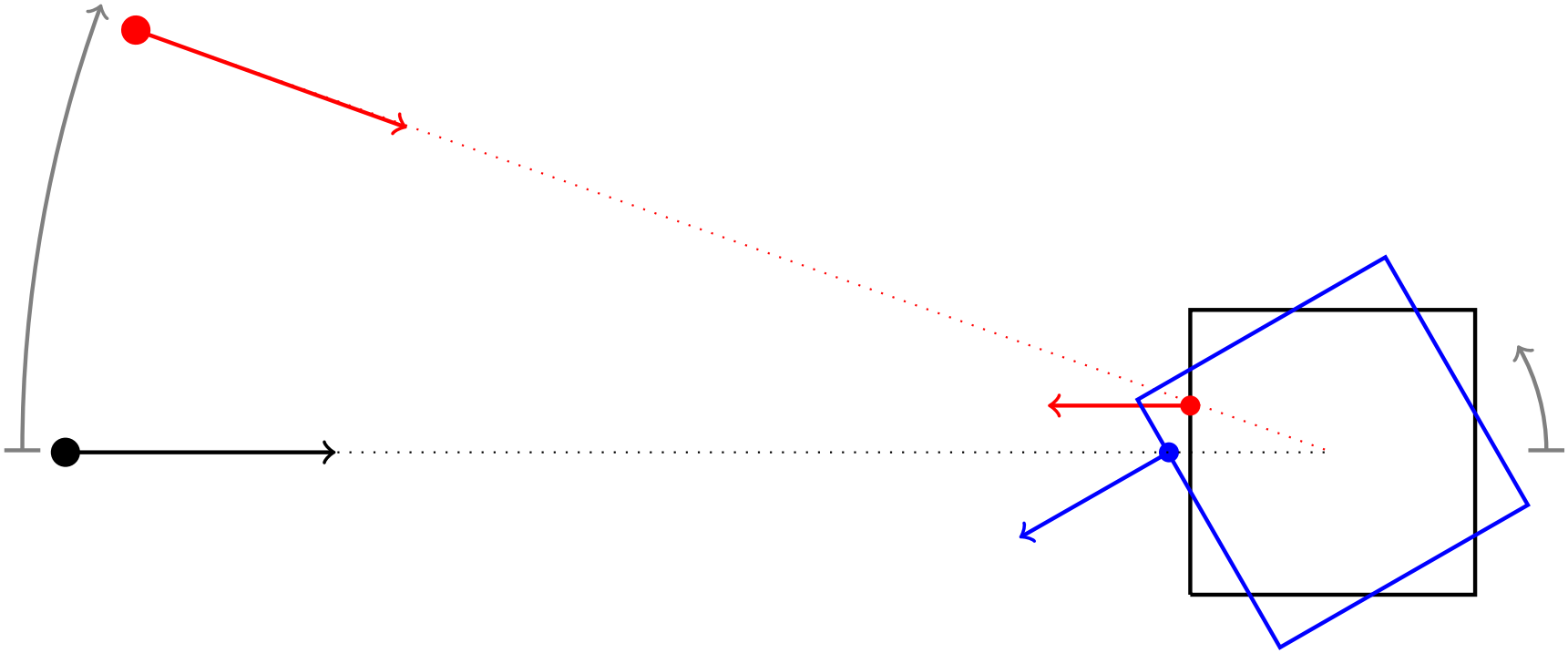


- Translate the ray point (subtract the object center).
- Rotate the ray point by the *inverse* rotation of the object.
- Rotate the ray vector by the *inverse* rotation of the object.
- Find t, the distance along the ray to the intersection.
- Use t with the original ray to find the point in world coordinates.

How do we find the normal?



How do we find the normal?



- Translate and rotate the ray to find the hit point.
- Use the *inverse* rotation of the object to rotate the ray.
- Find the normal at the hit point.
  Use the gradient as the quadric is untransformed.
- Rotate this normal using the *original* rotation matrix.

Putting it all together