Lecture 9: Parametric Forms and Ray Triangle Intersection

October 1, 2019

SageMath cs410lec09n01

localhost:8888/notebooks/CS410%20Fall2019/lectures/cs410lec09

Jupyter Notebook | cs410lec09n01

jupyter cs410lec09n01 Last Checkpoint: 2 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted SageMath 8.8

Intersecting 2D Parametric Lines

The following is important in its own right. However, it is also a solid lead up to understanding ray-triangle intersection.

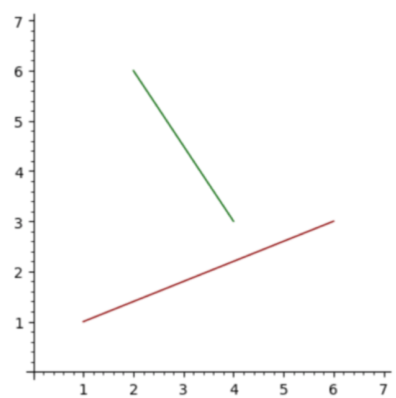
The goal here is to develop an intuition for parametric forms and then the geometry/algebra that answers questions such as "Do these to bounded line segments intersect"

Ross Beveridge, October 1, 2019

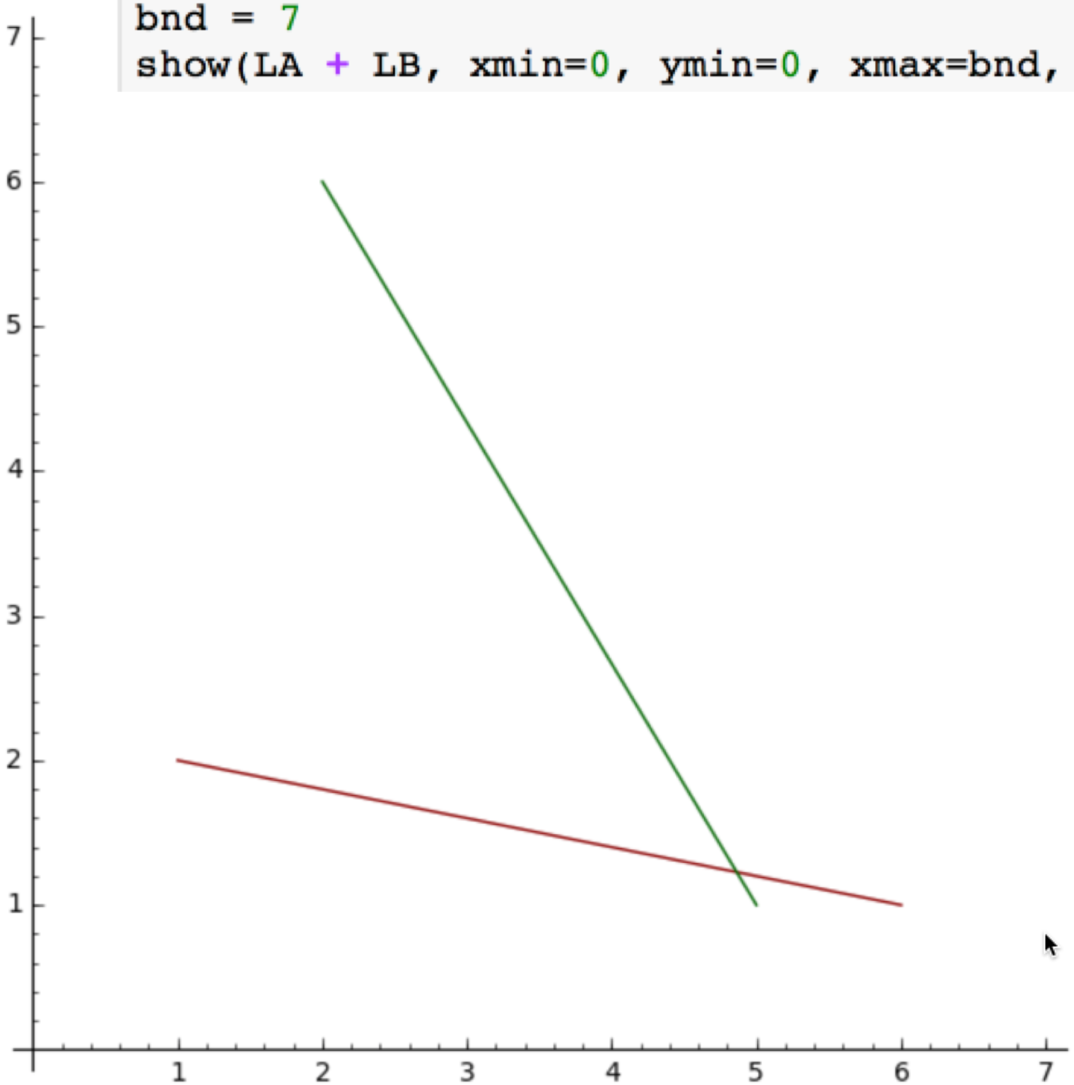
In [53]:
```
%display latex
```

To start, simply consider two bounded line segments between start and end points. This is also a good time to consider that two segments may well not intersect.

In [54]:
```
A1 = [1, 1]; A2 = [6, 3]
B1 = [2, 6]; B2 = [4, 3]
LA = line([A1,A2], color="darkred")
LB = line([B1,B2], color="darkgreen")
bnd = 7
show(LA + LB, xmin=0, ymin=0, xmax=bnd, ymax=bnd, aspect_ratio=1)
```



```
A1 = [1, 2]; A2 = [6, 1]
B1 = [2, 6]; B2 = [5, 1]
LA = line([A1,A2], color="darkred")
LB = line([B1,B2], color="darkgreen")
bnd = 7
show(LA + LB, xmin=0, ymin=0, xmax=bnd, ymax=bnd, aspect_ratio=1)
```



Surface the Parametric Form

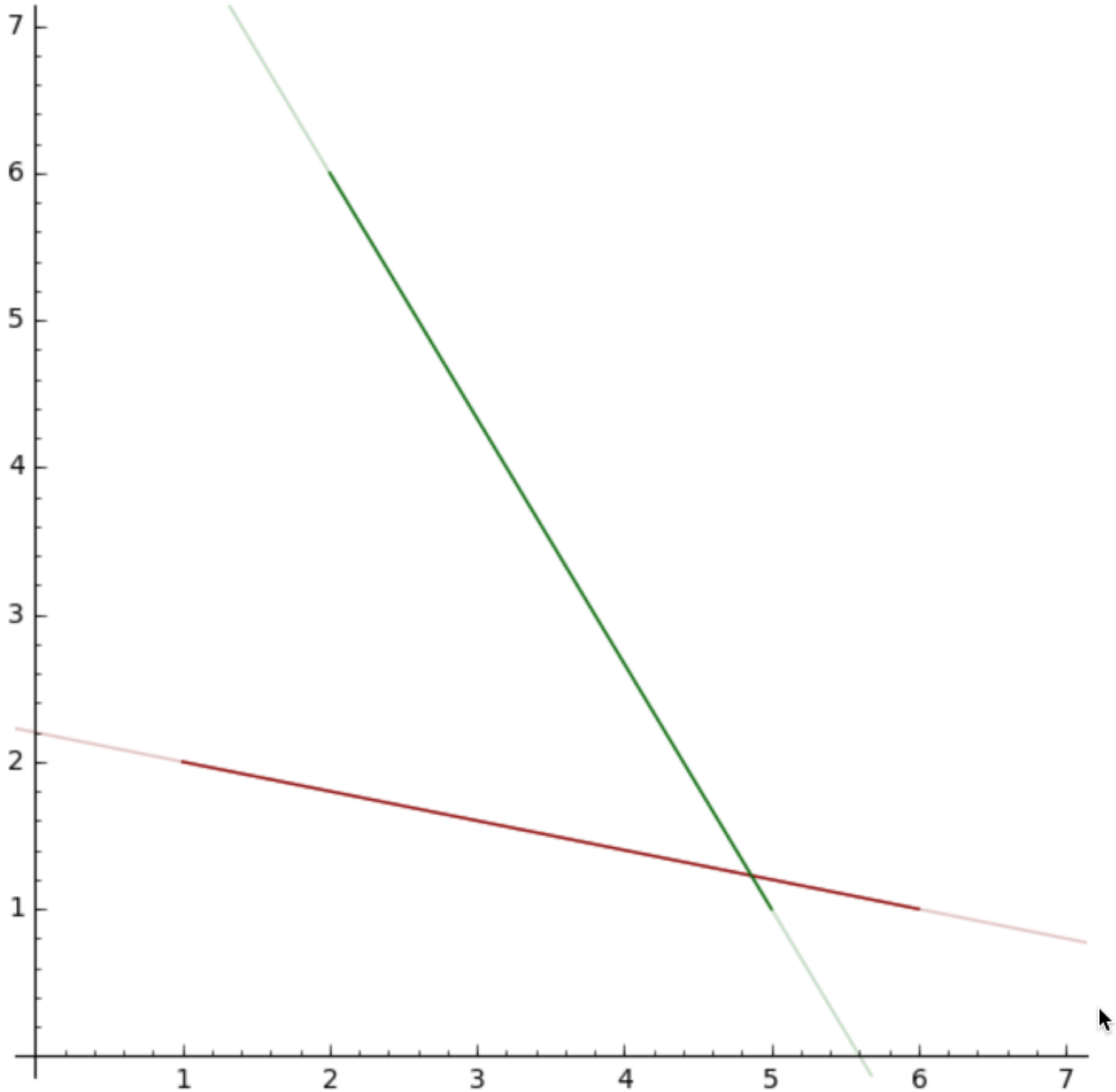
```
t = var('t')
def lapx(t) : return (A1[0]+(A2[0]-A1[0])*t)
def lapy(t) : return (A1[1]+(A2[1]-A1[1])*t)
def lbpx(t) : return (B1[0]+(B2[0]-B1[0])*t)
def lbpy(t) : return (B1[1]+(B2[1]-B1[1])*t)
LAP = parametric_plot((lapx(t),lapy(t)),(t,0.0,1.0),color='darkred')
LBP = parametric_plot((lbpx(t),lbpy(t)),(t,0.0,1.0),color='darkgreen')
bnd = 7
show(LAP + LBP, xmin=0, ymin=0, xmax=bnd, ymax=bnd, aspect_ratio=1)
```

The same thing written out in linear algebraic format.

$$\begin{vmatrix} x(t) \\ y(t) \end{vmatrix} = \begin{vmatrix} a1x \\ a1y \end{vmatrix} + \left(\begin{vmatrix} ax2 \\ ay2 \end{vmatrix} - \begin{vmatrix} a1x \\ a1y \end{vmatrix} \right) t$$

SageMath draws an identical figure.

Lines Know No Bounds

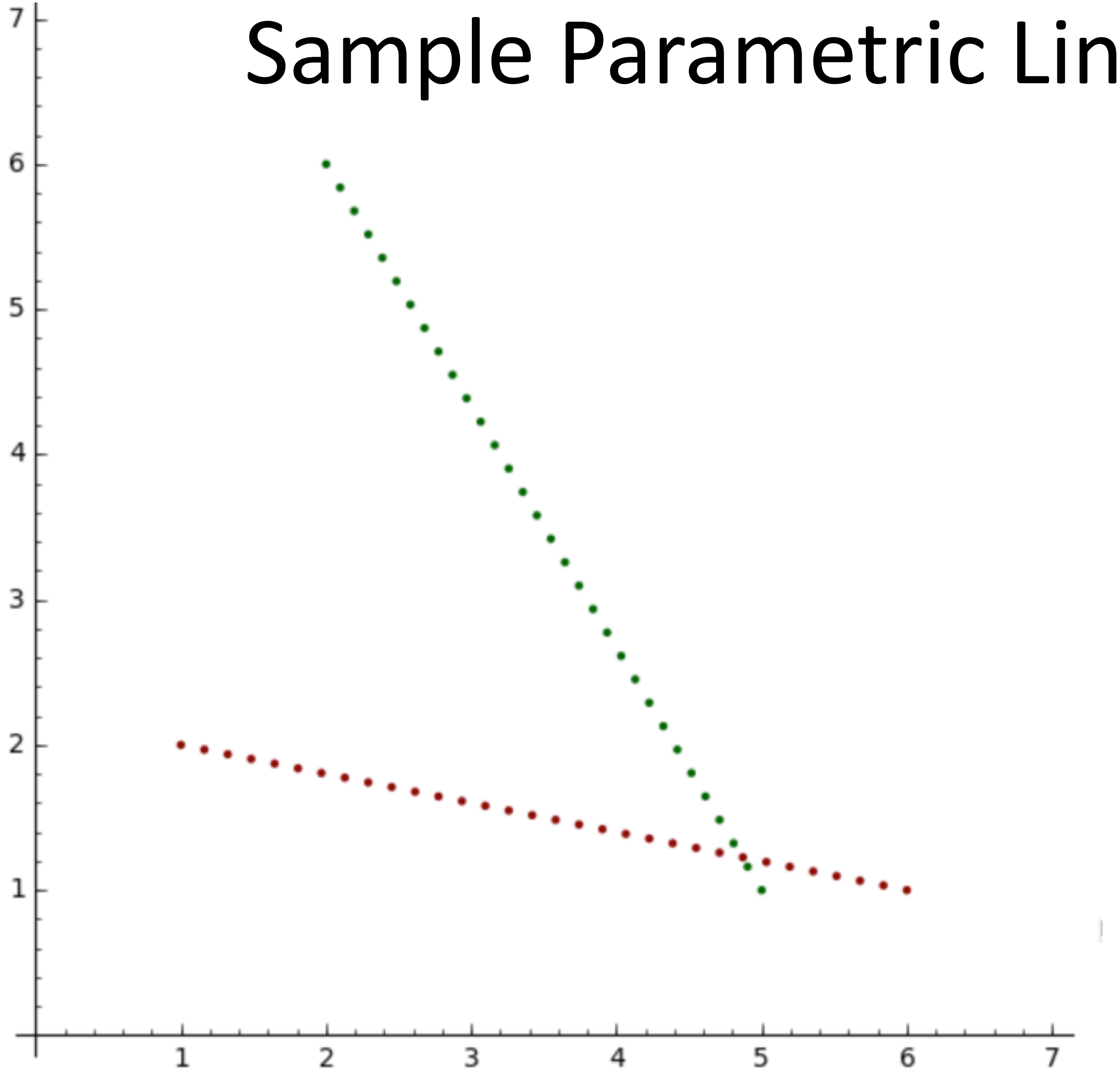


Any Value of *t* Will Do

```
LA = line([A1,A2], color="darkred")
LB = line([B1,B2], color="darkgreen")
LAP = parametric_plot((lapx(t),lapy(t)),(t,-10.0,10.0),color='darkred', alpha=0.25)
LBP = parametric_plot((lbpx(t),lbpy(t)),(t,-10.0,10.0),color='darkgreen', alpha=0.25)
bnd = 7
show(LA + LB + LAP + LBP, xmin=0, ymin=0, xmax=bnd, ymax=bnd, aspect_ratio=1)
```

- Consider how to draw an un-bounded line
- Not really possible
  - This is suggesting graphics concept of clipping
- But for the illustration, large bounds work
  - So notice t-values are beteen -10 and 10

Sample Parametric Line



Code to Sample

```
k   = 32.0
A1v = vector(A1); A2v = vector(A2);
B1v = vector(B1); B2v = vector(B2);
tss = [i/(k-1) for i in range(k)]
ptsa = [A1v + (A2v - A1v) * t for t in tss]
ptsb = [B1v + (B2v - B1v) * t for t in tss]
gptsa = [point(p,color='darkred') for p in ptsa]
gptsb = [point(p,color='darkgreen') for p in ptsb]
bnd = 7
show(sum(gptsa) + sum(gptsb), xmin=0, ymin=0, xmax=bnd, ymax=bnd, aspect_ratio=1)
```

- Python is making this 'easy'
- How, by allowing enumeration of list elements
- Here 32 points are created that are evenly sampled along the two line segments

How About Intersection

- Pair of equations in two unknowns

$$-(ax_1 - ax_2)t + ax_1 = -(bx_1 - bx_2)s + bx_1$$

$$-\left(ay_1 - ay_2\right)t + ay_1 = -\left(by_1 - by_2\right)s + by_1$$

- We will shortly consider turning this into a matrix inversion problem, but not yet.

Solution

- Courtesy of SageMath solve command

$$s = -\frac{ax_2\left(ay_1 - by_1\right) - ax_1\left(ay_2 - by_1\right) - \left(ay_1 - ay_2\right)bx_1}{\left(ay_1 - ay_2\right)bx_1 - \left(ay_1 - ay_2\right)bx_2 - ax_1\left(by_1 - by_2\right) + ax_2\left(by_1 - by_2\right)}$$

$$t = \frac{\left(ay_1 - by_2\right)bx_1 - \left(ay_1 - by_1\right)bx_2 - ax_1\left(by_1 - by_2\right)}{\left(ay_1 - ay_2\right)bx_1 - \left(ay_1 - ay_2\right)bx_2 - ax_1\left(by_1 - by_2\right) + ax_2\left(by_1 - by_2\right)}$$

- For our specific example

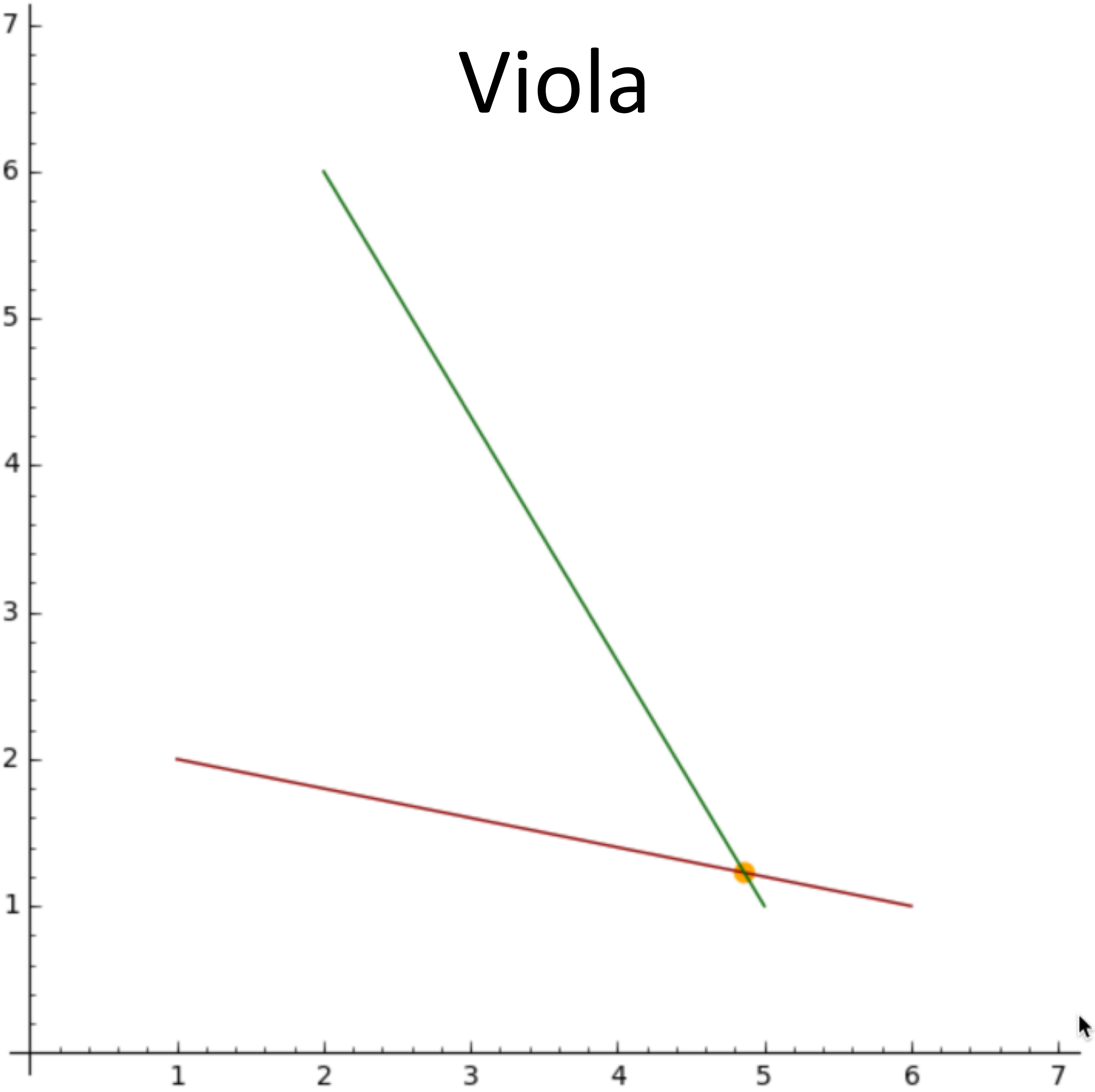
$$\left[s = \left(\frac{21}{22}\right), t = \left(\frac{17}{22}\right)\right]$$

Now Draw It

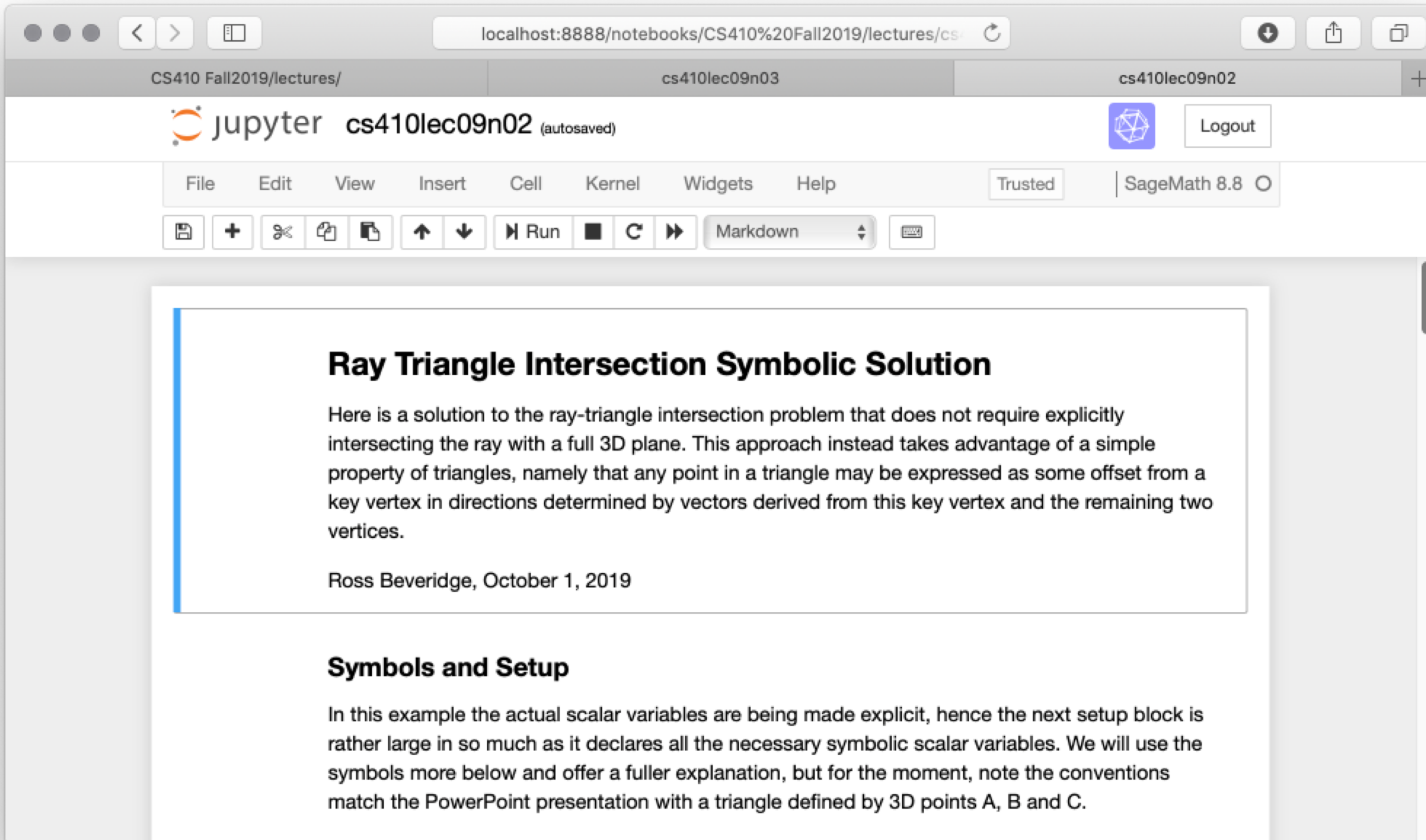
```
ax1  = A1[0]; ax2 = A2[0]; ay1 = A1[1]; ay2 = A2[1]
bx1  = B1[0]; bx2 = B2[0]; by1 = B1[1]; by2 = B2[1]
eq1v = ax1 + (ax2 - ax1) * t == bx1 + (bx2 - bx1) * s
eq2v = ay1 + (ay2 - ay1) * t == by1 + (by2 - by1) * s
resv = solve([eq1v,eq2v], s, t)
```

```
tstar = resv[0][1].rhs()
LA  = line([A1,A2], color="darkred")
LB  = line([B1,B2], color="darkgreen")
poi = point((lapx(tstar), lapy(tstar)),size=64,color='orange')
bnd = 7
show(LA + LB + poi, xmin=0, ymin=0, xmax=bnd, ymax=bnd, aspect_ratio=1)
```

Viola



New Topic: Ray Triangle Intersection



Ray Triangle Intersection Symbolic Solution

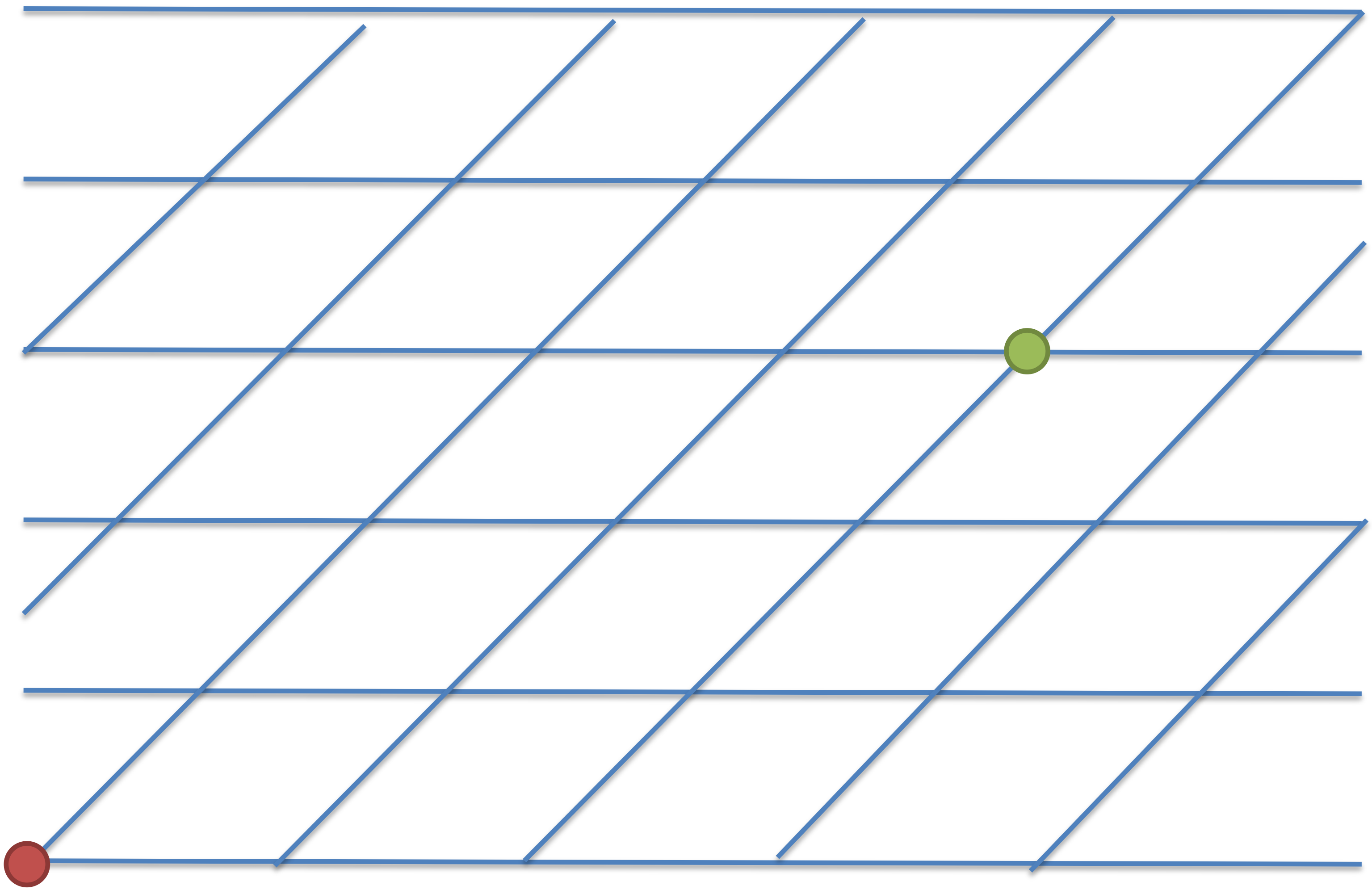
Here is a solution to the ray-triangle intersection problem that does not require explicitly intersecting the ray with a full 3D plane. This approach instead takes advantage of a simple property of triangles, namely that any point in a triangle may be expressed as some offset from a key vertex in directions determined by vectors derived from this key vertex and the remaining two vertices.

Ross Beveridge, October 1, 2019

Symbols and Setup

In this example the actual scalar variables are being made explicit, hence the next setup block is rather large in so much as it declares all the necessary symbolic scalar variables. We will use the symbols more below and offer a fuller explanation, but for the moment, note the conventions match the PowerPoint presentation with a triangle defined by 3D points A, B and C.

Triangle Warm-up



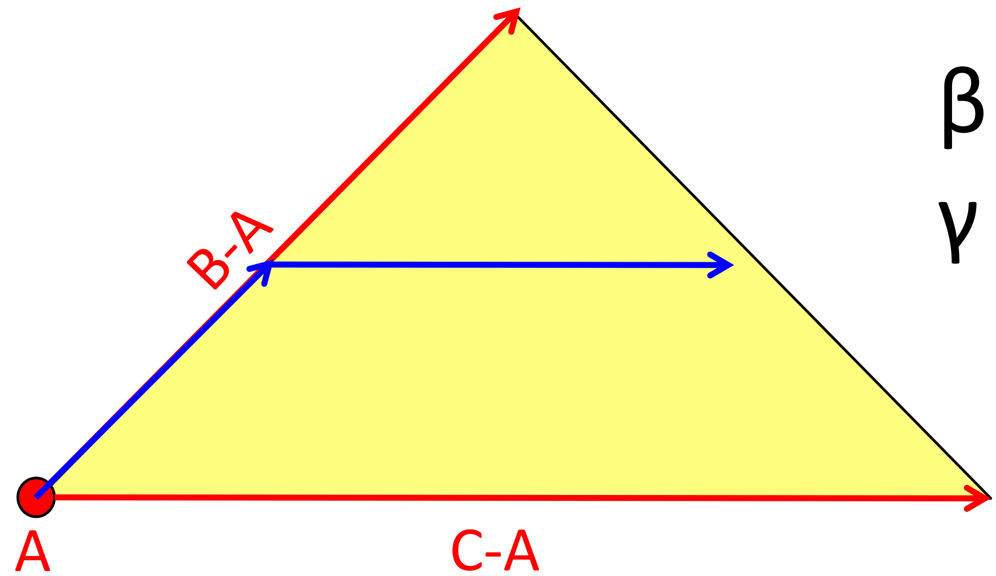
How do you 'drive' from the red to the green point?

Ray/Triangle Intersections

- Ray/Triangle intersections are efficient and can be computed directly in 3D
  - No need for ray/plane intersection
- Solution relies on the following implicit definition of a triangle:

$$P = A + \beta(B - A) + \gamma(C - A)$$
$$\beta \geq 0, \gamma \geq 0, \beta + \gamma \leq 1$$

Implicit Triangles



$\beta = 0.5$

$\gamma = 0.48$

Triangle Parametric Form

- There are two free parameters
- They select points inside the triangle

  ... and outside it as well!

$$P(\beta, \gamma) = A + \beta(B - A) + \gamma(C - A)$$

$$P(\beta, \gamma) = = \begin{bmatrix} -(ax - bx)\beta - (ax - cx)\gamma + ax \\ -(ay - by)\beta - (ay - cy)\gamma + ay \\ -(az - bz)\beta - (az - cz)\gamma + az \end{bmatrix}$$

Find Ray Plane Intersection

- If they intersect, there is a solution to:

$$\begin{bmatrix} dxt + lx \\ dyt + ly \\ dzt + lz \end{bmatrix} = \begin{bmatrix} -(ax - bx)\beta - (ax - cx)\gamma + ax \\ -(ay - by)\beta - (ay - cy)\gamma + ay \\ -(az - bz)\beta - (az - cz)\gamma + az \end{bmatrix}$$

$$\begin{bmatrix} dxt + lx \\ dyt + ly \\ dzt + lz \end{bmatrix} + \begin{bmatrix} (ax - bx)\beta + (ax - cx)\gamma - ax \\ (ay - by)\beta + (ay - cy)\gamma - ay \\ (az - bz)\beta + (az - cz)\gamma - az \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Standard 3x3 Linear System

- Rearranging terms we have a standard:

$$M\,X = Y$$

- Expanded this is …

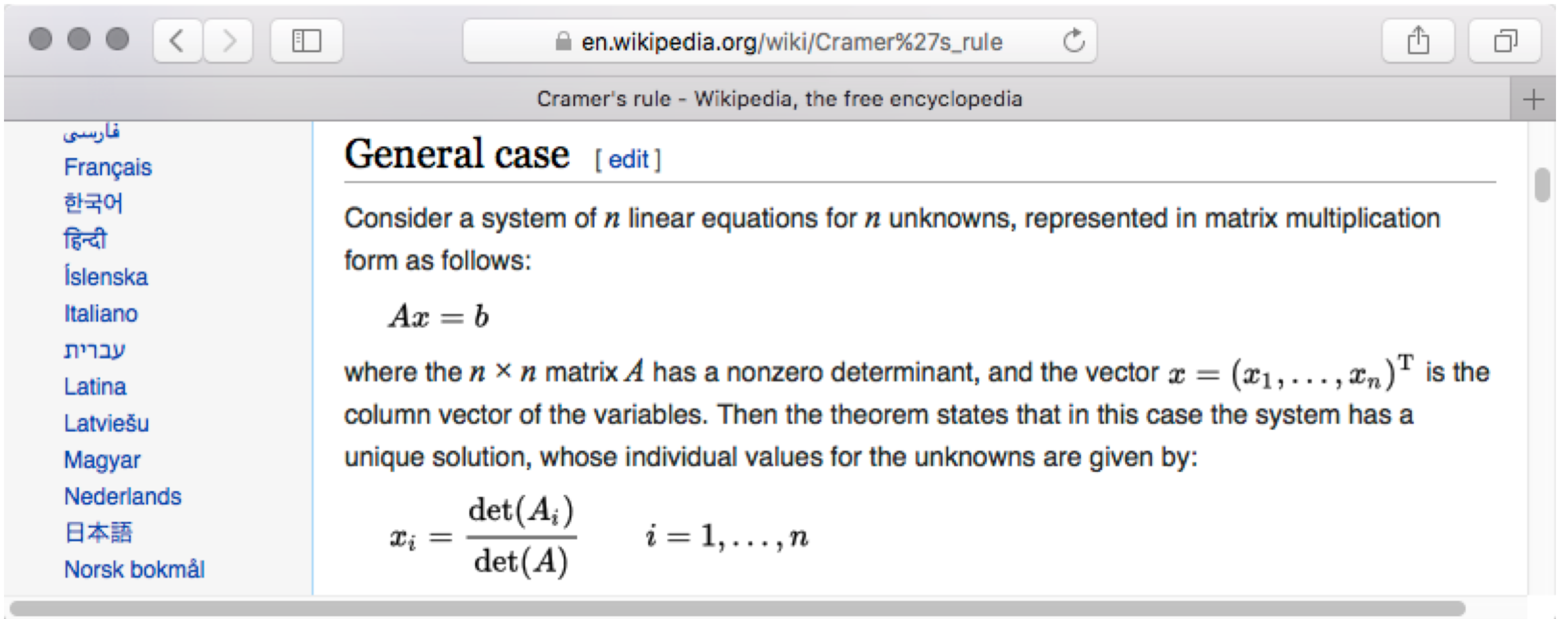
$$\begin{bmatrix} ax - bx & ax - cx & dx \\ ay - by & ay - cy & dy \\ az - bz & az - cz & dz \end{bmatrix} * \begin{bmatrix} \beta \\ \gamma \\ t \end{bmatrix} = \begin{bmatrix} ax - lx \\ ay - ly \\ az - lz \end{bmatrix}$$

Solve for Intersection

- Using favorite linear system method
  - More on this soon
- What if the matrix is singular?
  - then the plane doesn't intersect the ray
- The point is inside the triangle and in front of the camera if and only if
  - $\beta \geq 0$
  - $\Upsilon \geq 0$
  - $\beta + \Upsilon \leq 1$
  - $t > 0$
  - Note: Knowing t yields the point of intersection

Using Cramer’s Rule

- One approach users Cramer’s Rule



en.wikipedia.org/wiki/Cramer%27s_rule

Cramer's rule - Wikipedia, the free encyclopedia

فارسی
Français
한국어
हिन्दी
Íslenska
Italiano
עברית
Latina
Latviešu
Magyar
Nederlands
日本語
Norsk bokmål

General case [edit]

Consider a system of n linear equations for n unknowns, represented in matrix multiplication form as follows:

$$Ax = b$$

where the $n \times n$ matrix A has a nonzero determinant, and the vector $x = (x_1, \ldots, x_n)^{\mathrm{T}}$ is the column vector of the variables. Then the theorem states that in this case the system has a unique solution, whose individual values for the unknowns are given by:

$$x_i = \frac{\det(A_i)}{\det(A)} \qquad i = 1, \ldots, n$$

The efficiency of this approach compared to a numerical package depends upon details, including the care taken implementing the actual code. For example, using early exit strategies.

Here In Full Glory

$$\mathtt{M} = \begin{bmatrix} ax-bx & ax-cx & dx \\ ay-by & ay-cy & dy \\ az-bz & az-cz & dz \end{bmatrix}$$

$$M_1 = \begin{bmatrix} ax-lx & ax-cx & dx \\ ay-ly & ay-cy & dy \\ az-lz & az-cz & dz \end{bmatrix}, \quad M_2 = \begin{bmatrix} ax-bx & ax-lx & dx \\ ay-by & ay-ly & dy \\ az-bz & az-lz & dz \end{bmatrix}, \quad M_3 = \begin{bmatrix} ax-bx & ax-cx & ax-lx \\ ay-by & ay-cy & ay-ly \\ az-bz & az-cz & az-lz \end{bmatrix}$$

$$\beta = \frac{|M_1|}{|M|} = \frac{((az-cz)dy-(ay-cy)dz)(ax-lx)-((az-cz)dx-(ax-cx)dz)(ay-ly)+((ay-cy)dx-(ax-cx)dy)(az-lz)}{((az-cz)dy-(ay-cy)dz)(ax-bx)-((az-cz)dx-(ax-cx)dz)(ay-by)+((ay-cy)dx-(ax-cx)dy)(az-bz)}$$

$$\gamma = \frac{|M_2|}{|M|} = \frac{((az-lz)dy-(ay-ly)dz)(ax-bx)-((az-lz)dx-(ax-lx)dz)(ay-by)+((ay-ly)dx-(ax-lx)dy)(az-bz)}{((az-cz)dy-(ay-cy)dz)(ax-bx)-((az-cz)dx-(ax-cx)dz)(ay-by)+((ay-cy)dx-(ax-cx)dy)(az-bz)}$$

$$t = \frac{|M_3|}{|M|} = \frac{((ay-ly)(az-cz)-(ay-cy)(az-lz))(ax-bx)-((ax-lx)(az-cz)-(ax-cx)(az-lz))(ay-by)+((ax-lx)(ay-cy)-(ax-cx)(ay-ly))(az-bz)}{((az-cz)dy-(ay-cy)dz)(ax-bx)-((az-cz)dx-(ax-cx)dz)(ay-by)+((ay-cy)dx-(ax-cx)dy)(az-bz)}$$

Without serious effort to collect terms this solution will run slower than a numerical solver.

However, collecting terms is not that difficult.

SageMath Worked Example

jupyter cs410lec09n03 Last Checkpoint: 09/18/2018 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | SageMath 8.8

Run Code

Ray Triangle Intersection Example

Here is worked numerical example of Ray Triangle intersection with a 3D visualization of the process and result.

Ross Beveridge, October 1, 2019

The actual 3D position of the 3 triangle corners as well as the ray starting point and direction are given here at the head of the file. Initially the example is setup using these values:

Av = vector(SR, 3, (3,0,0)); Bv = vector(SR, 3, (0,3,0)); Cv = vector(SR, 3, (0,0,3)); Lv = vector(SR, 3, (0,0,0)); Dv = vector(SR, 3, (1,1,1));

To further explore consider these alternatives:

Av = vector(SR, 3, (6,0,0)); Bv = vector(SR, 3, (0,6,0)); Cv = vector(SR, 3, (0,0,6)); Lv = vector(SR, 3, (1,1,1)); Dv = vector(SR, 3, (1,1,1));

Av = vector(SR, 3, (6,0,0)); Bv = vector(SR, 3, (0,6,0)); Cv = vector(SR, 3, (0,0,6)); Lv = vector(SR, 3, (1,1,1)); Dv = vector(SR, 3, (1,0,0));

Av = vector(SR, 3, (6,0,0)); Bv = vector(SR, 3, (0,6,0)); Cv = vector(SR, 3, (0,0,6)); Lv = vector(SR, 3, (0,0,0)); Dv = vector(SR, 3, (0,1,1));

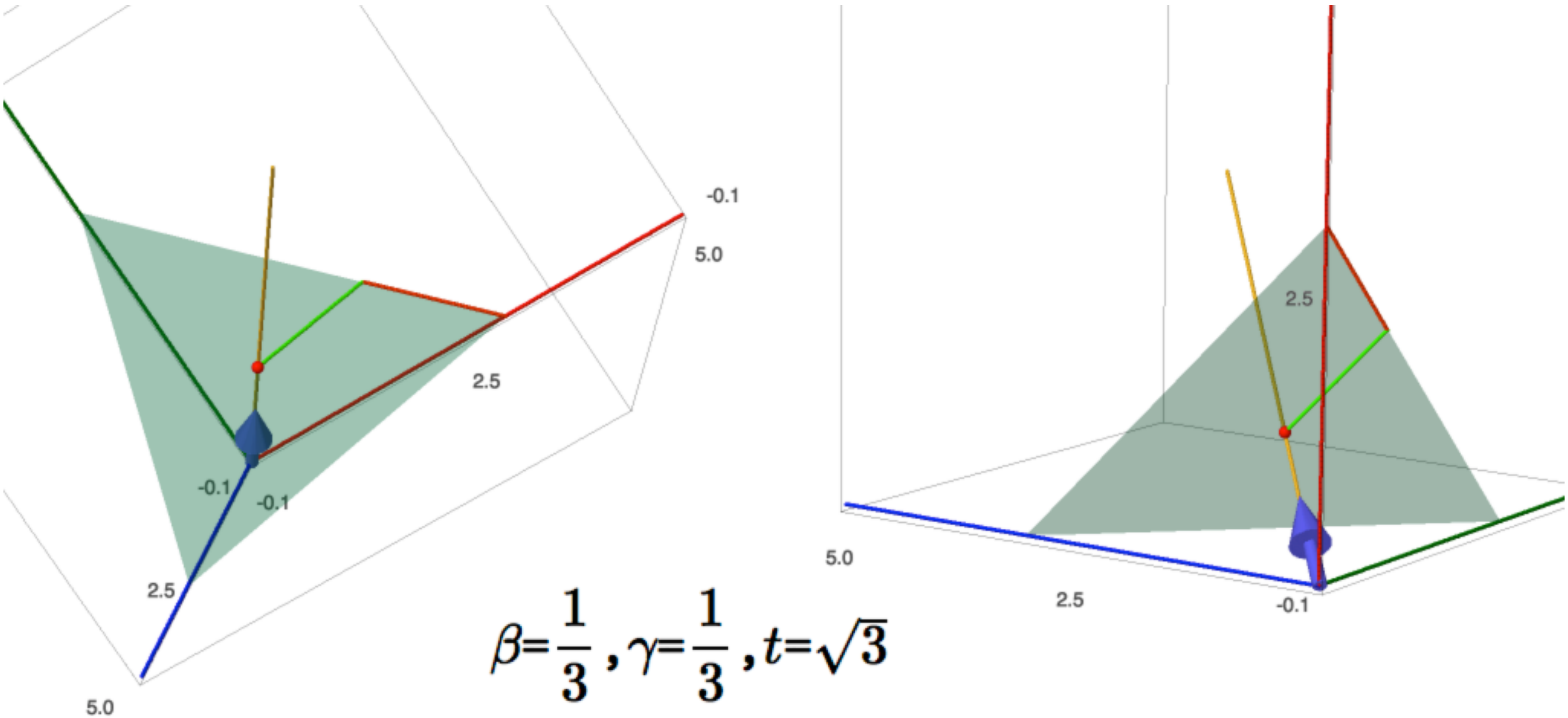
Problem / Opportunity

- You have two ways to compute the same three values.
- One, at least, is rife with chances to make errors when being converted to code.
- But, that same option holds promise of efficiency through early termination.

How would one write code to confidently debug and test a relatively complicated geometric computation …

Example 1 Visualization

Ray from origin in direction (1,1,1) with triangle pinned at 3 out each of the X, Y and Z axes.



$$\beta=\frac{1}{3},\gamma=\frac{1}{3},t=\sqrt{3}$$