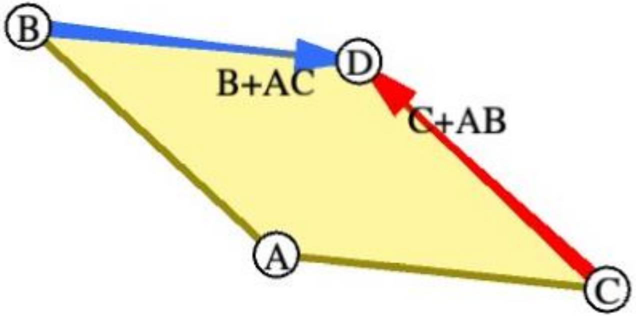
PLAYING WITH POINTS AND VECTORS



CS3451 FALL 2020

Alexander GOEBEL

PHSE 1: Problem statement

The purpose of phase 1 is to complete a parallelogram given a set of three of its vertices {A,B,C} by locating its fourth vertex {D}.

COMMENTS:

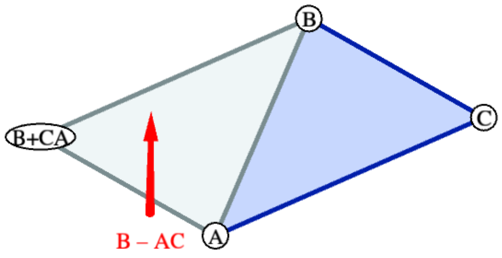
The fourth vertex location can have multiple (3) unique solutions obtained by (6) different methods.

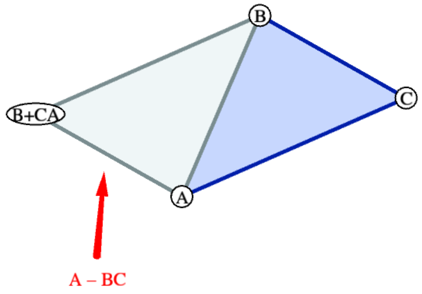
The set {A,B,C} is assumed to not be colinear.

PHSE 1: Solution outline

I used the Point + Vector construct using the point C and the vector AB initially, then I functionally reimplemented the Point + Vector for each edge of the set {A,B,C} and its respective point.

- C + AB, C – AB, A + BC, A – BC, B + AC, B - AC
- In practice, only 3 solutions were needed, as functionally B – AC (aka B + CA) was equivalent to A – BC, A + BC was equivalent to C – AB, and B + AC was equivalent to C + AB





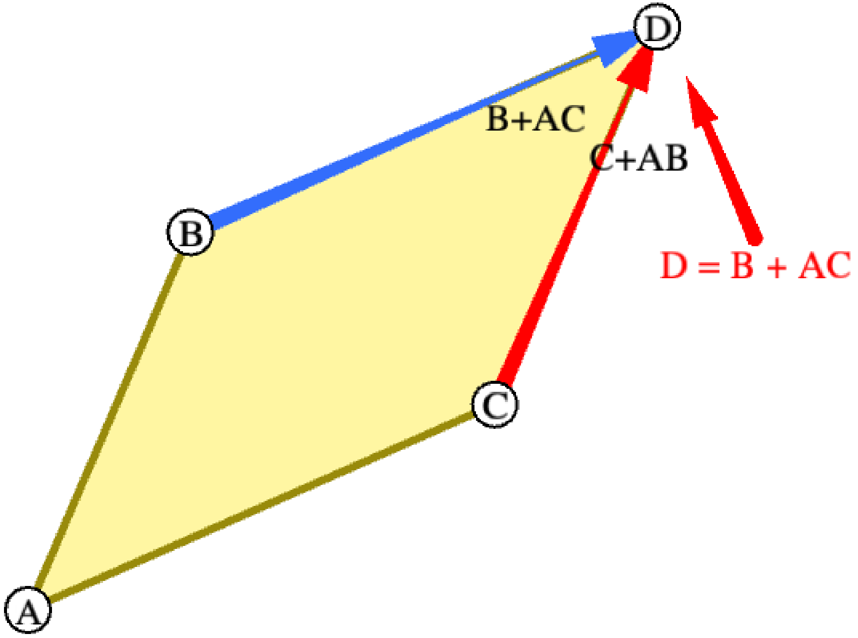
PHSE 1: Solution math

D = B + AC

JUSTIFICATION:

A parallelogram {A,B,C,D},
has vector equality: AC = BD.

Hence, D – B = AC, and D = B + AC.

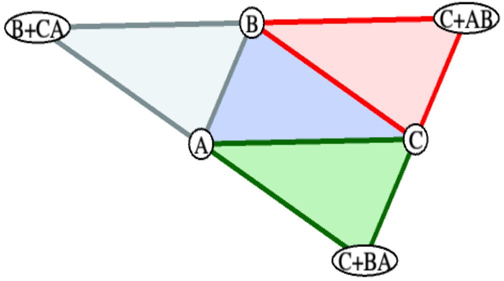


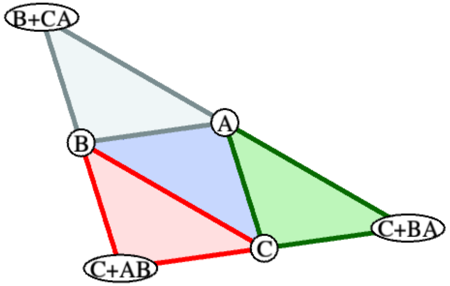
PHSE 1: Solution examples and limitations

There are 6 possible (3 unique) solutions.

My solution works for both:

CW orientation of {A,B,C} & CCW orientation of {A,B,C}





PHSE 1: Code

```
//========================================================================= PART 1
boolean showJ=false, showK=false, showL=false, showQ=false;
void showPart1(PNT A, PNT B, PNT C, PNT D) //
  {
  PartTitle[1] =  "Complete the parallelogram"; // https://en.wikipedia.org/wiki/Parallelogram

  VCT AB = V(A,B);
  if (showQ) { //Show singular, solid parallelogram if 'q' pressed
    PNT J = P(C,V(A,B)); // obtain point D from C + AB
    cwfo(dgold,5,gold,100); showLoop(A, B, J, C); //show solid parallelogram with dgold border and gold fill
    show(C,V(A,B),red,"C+AB"); show(B,V(A,C),blue,"B+AC"); //show two methods that could be used to obtain this solution
    circledLabel(J,"D"); // label point D
  } else {cwfo(dblue,5,blue,70); showLoop(A, B, C);} // show given triangle {A,B,C} with dblue border and blue fill
  if (showJ) {PNT J = P(C,V(A,B)); cwfo(dred,5,pink,70); showLoop(B, C, J); J.circledLabel("C+AB");} //Show parallelogram for Point D from C + AB
  if (showK) {PNT K = P(C,V(B,A)); cwfo(dgreen,5,green,70); showLoop(A, C, K); K.circledLabel("C+BA");} //Show parallelogram for Point D from C - AB
  if (showL) {PNT L = P(B,V(C,A)); cwfo(dmetal,5,metal,70); showLoop(A, B, L); L.circledLabel("B+CA");} // Show  Point D from B - CA


  guide="Part 1 keys: (j/k/l/q) to show/hide solutions";
  A.circledLabel("A"); B.circledLabel("B"); C.circledLabel("C");
  }
```

PHSE 1: Sources

My solution is on the fact that AC = BD in a parallelogram {A,B,C,D} that AC = BD, as outlined by the two properties of parallelogram:

- Two pairs of opposite sides are parallel (by definition).
- Two pairs of opposite sides are equal in length.

These two properties were found from reference by Wikipedia at https://en.wikipedia.org/wiki/Parallelogram

PHSE 2: Problem statement

Compute the Fermat point of a triangle given its three vertices {A, B,C}.

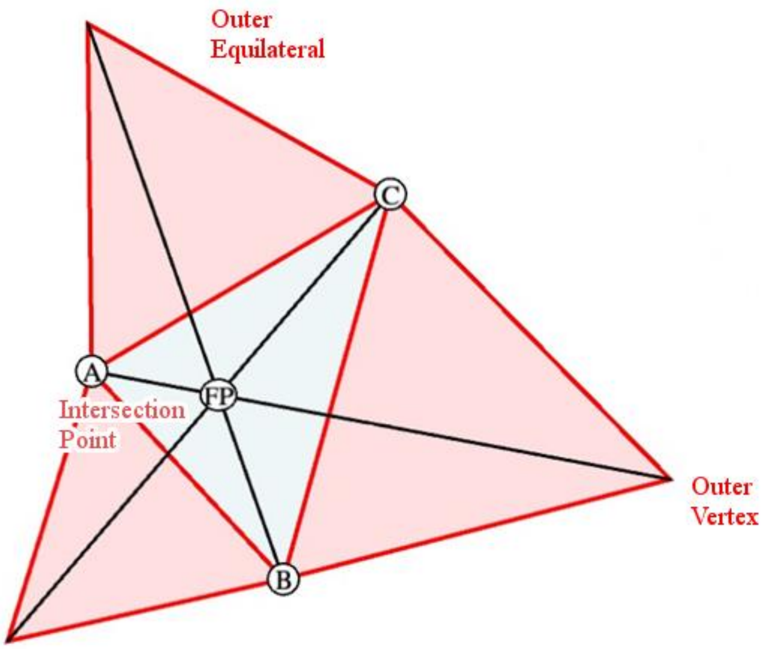
COMMENTS:

The vertices {A,B,C} can form a triangle with either:

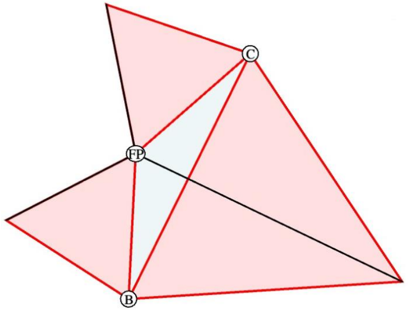
- An inner angle exceeding 120 degrees
- Only having inner angles that are at most 120 degrees

PHSE 2: Solution outline

We compute the Fermat point of a triangle as the intersection of any two lines drawn from the outer vertices of the equilateral triangles formed sharing each of the edges of the original triangle.



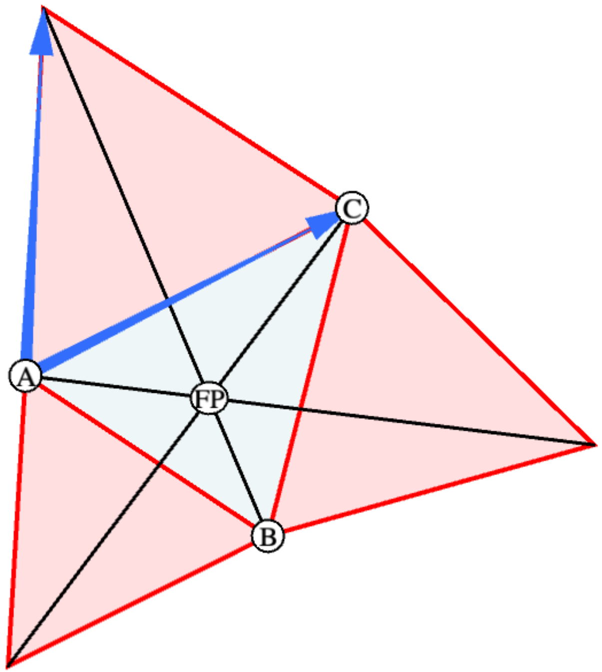
In the case that one of the triangle's angles exceeded 120 degrees, the Fermat point was located at the vertex of that angle.



PHSE 2: Solution math

Outer equilateral triangle formed:

- Rotating an edge (example AC) by $\pi/3$ radians out from the original triangle
  - Would be equilateral because all sides share the same length and same angle ($\pi/3$)
- Draw line from outer vertex of equilateral to opposing angle of the inner (original) triangle.
- Intersection is by definition the Fermat point



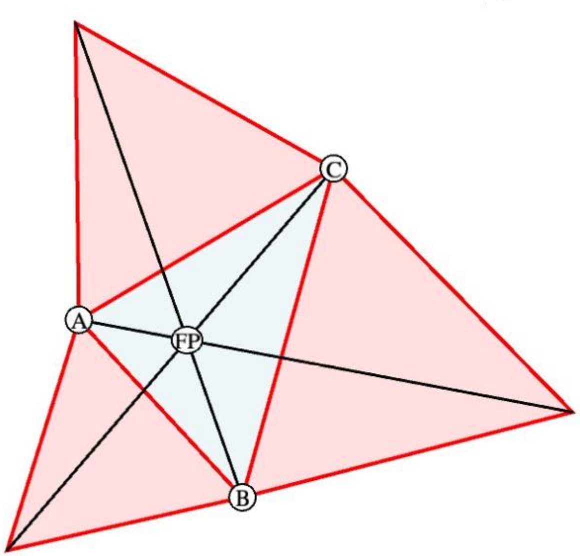
JUSTIFICATION:

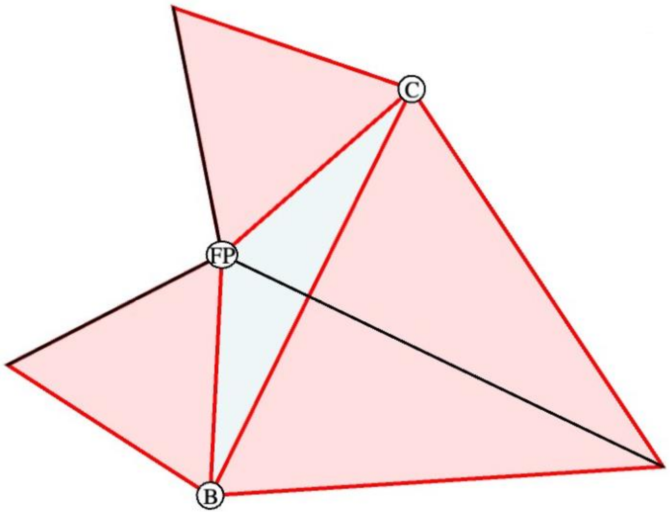
Each edge of triangle rotated helps determine equilateral outer triangle because:

- Would be equilateral because all sides share the same length and same angle ($\pi/3$)

PHSE 2: Solution examples and limitations

My solution correctly models the Fermat point when all inner angles of the triangle ABC are at most 120 degrees and also when one of the angles exceeds 120 degrees.





PHSE 2: Code

```
//======================================================================= PART 3
void showPart3(PNT A, PNT B, PNT C, PNT D) //
  {
  PartTitle[3] = "Fermat point"; // https://en.wikipedia.org/wiki/Fermat_point

  cwfo(dmetal,4,metal,70); showLoop(A,B,C); //Draw triangle from given points with dark metal border and metal fill

  VCT AB = V(A,B); VCT AC = V(A,C); VCT BC = V(B,C); //Edges of the original triangle
  PNT ABV = (angle(AC,AB) > 0) ? P(B,R(AB,2*PI/3.0)) : P(A,R(AB,5*PI/3.0)); //obtains third vertex of AB edge
  PNT ACV = (angle(AC,AB) > 0) ? P(A,R(AC,5*PI/3.0)) : P(C,R(AC,2*PI/3.0)); //obtains third vertex of AC edge
  PNT BCV = (angle(AB,AC) > 0) ? P(B,R(BC,5*PI/3.0)) : P(C,R(BC,2*PI/3.0)); //obtains third vertex of BC edge

  cwfo(dred,4,pink,70); //Border dark red, fill pink
  showLoop(A,ABV,B); showLoop(A,ACV,C); showLoop(B,BCV,C); //Draw outer equilateral triangles using above coloring
  if (angle(AB,AC) > 2*PI/3 || angle(AB,AC) < -2*PI/3)  //Detects if angle between AB and AC is greater than 120
    {
      cwf(black,3,black);
      show(ABV,A); show(ACV,A); show(BCV,A); //Draw black lines from outer vertices to obtuse angle (C)

      B.circledLabel("B"); C.circledLabel("C"); A.circledLabel("FP"); //FP for Fermat Point
    }
  else if (angle(V(B,A),V(B,C)) > 2*PI/3 || angle(V(B,A),V(B,C)) < -2*PI/3) //Detects if angle between BA and BC is greater than 120
    {
      cwf(black,3,black);
      show(ABV,B); show(ACV,B); show(BCV,B); //Draw black lines from outer vertices to obtuse angle (B)

      A.circledLabel("A"); C.circledLabel("C"); B.circledLabel("FP"); //FP for Fermat Point
    }
  else if (angle(V(C,A),V(C,B)) > 2*PI/3 || angle(V(C,A),V(C,B)) < -2*PI/3) //Detects if angle between CA and CB is greater than 120
    {
      cwf(black,3,black);
      show(ABV,C); show(ACV,C); show(BCV,C); //Draw black lines from outer vertices to obtuse angle (C)

      A.circledLabel("A"); B.circledLabel("B"); C.circledLabel("FP"); //FP for Fermat Point
    }
  else
    {
      cwf(black,3,black);
      show(ABV,C); show(ACV,B); show(BCV,A); //Draw lines from outer vertices to inner angles

      /*PNT FP = P(((ABV.x*C.y-ABV.y*C.x)*(A.x-BCV.x)-(ABV.x-C.x)*(A.x*BCV.y-A.y*BCV.x))/((ABV.x-C.x)*(A.y-BCV.y)-(ABV.y-C.y)*(A.x-BCV.x)),
        ((ABV.x*C.y-ABV.y*C.x)*(A.y-BCV.y)-(ABV.y-C.y)*(A.x*BCV.y-A.y*BCV.x))/((ABV.x-C.x)*(A.y-BCV.y)-(ABV.y-C.y)*(A.x-BCV.x))); //https://en.wikipedia.org/wiki/Line%E2%80%93line_intersection#
      FP.circledLabel("FP"); //FP for Fermat Point*/

      VCT QnP = V(BCV,ABV); //ABV-BCV
      VCT R = V(BCV,A); //Vector from BCV to A
      VCT S = V(ABV,C); //Vector from ABV to C
      double num = QnP.x*S.y-QnP.y*S.x; // (ABV-BCV) cross S
      double den = R.x*S.y-R.y*S.x; // R cross S
      float t = (float) num/ (float) den; //proportion of R from the equation p + tr
      VCT tR = V(t,R); //scaled R vector
      PNT FPalt = P(BCV,tR);//point at the end of scaled R vector from BCV (The Fermat Point)

      A.circledLabel("A"); B.circledLabel("B"); C.circledLabel("C");
      FPalt.circledLabel("FP");//https://stackoverflow.com/questions/563198/how-do-you-detect-where-two-line-segments-intersect
    }
  }
```

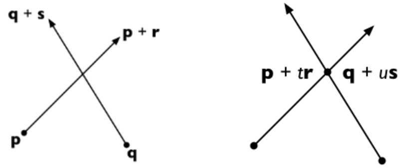
PHSE 2: Sources

My solution is based on the fact the Fermat point of a triangle is described by two cases of the given triangle, in which:

- (1) The is an inner angle between two of the edges of the triangle greater than 2pi/3 radians. In this case, the Fermat point is at the vertex located between those edges
- (2) None of the triangle's inner angles are greater than 2pi/3 radians. In this case, the Fermat point is described as the intersection between any two lines formed from the outer vertices of the three equilateral triangles sharing an edge with original triangle to the opposing vertices in the original triangle of the shared edges.

This understanding was obtained from Wikipedia at https://en.wikipedia.org/wiki/Fermat_point

My solution is additionally based on the formula for obtaining the intersection point:



$$p + \frac{(q - p) \times s}{(r \times s)}\, r \;=\; \text{Intersection Point}$$

As described on Stack Overflow

https://stackoverflow.com/questions/563198/how-do-you-detect-where-two-line-segments-intersect