

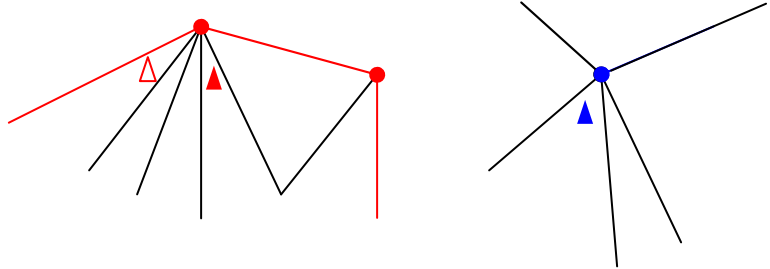
# Prevent flip & overlap

Ang Lee [coolpix@cc.gatech.edu](mailto:coolpix@cc.gatech.edu)

I wrote a function, when given the (X,Y) coordinate of the moving tip (on the border, red case below), or a existing vertex(either on the border or inside the mesh, blue case below), and its index in G-table, the function will return flip happened or not.

```
bool flip(float X, float Y, int Index_in_Gtable)
{
```

```
    int i;
    int firstCorner;
    int lasti = -1;
    bool IsOnBorder = 0;
    for (i=0; i<ccount; i++)
    {
        if (v[i] == Index_in_Gtable)
        {
            firstCorner = i;
            break;
        }
    }
    //after this 'i' contain a corner whose vertex is Index_in_Gtable
```



In general, there could be two cases: first, the corner found is on the border (red ▲ case), second, the corner is incident to a vertex which is not on the border (blue ▲ case)

```
    do{
        lasti = i;
        if(o[p(i)] != -1)
            i = p(o[p(i)]);
        if (i == lasti ) // if it is a vertex not on the border, the incident corner do not form a loop, and i can
                        // never be last i
        {
            // printf("it must be a vertex on the border");
            IsOnBorder = 1;
            break;
        }
    }
    while (i != firstCorner);
```

After the above do while loop, in red case, the corner 'i' will be a corner incident at the given border vertex, and no left neighbor (▲). This was done by keep moving ▲ left (to ▲.p.o.p), until there is no left triangle ▲.p.o = -1. (If there is only one triangle incident at the vertex, for example, a new created vertex ▲ just stays.) A blue case is found when keep moving left and finally wrap back to the first corner, after wrap back, go on.

In red case, we test if the user is moving the tip in to a triangle or not.

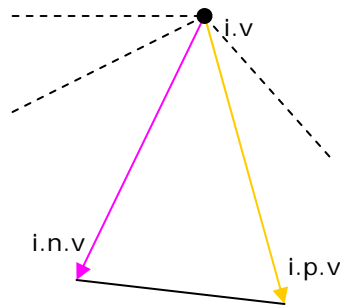
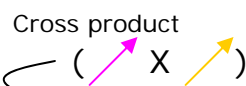
Function "InTriangle(X, Y, Index\_in\_Gtable)" return true if the moving vertex moves into the triangles which DO NOT have Index\_in\_Gtable be one of its three vertices. It is reasonable to let the use move in to the triangles which incident at the moving vertex (whose index in G-table is Index\_in\_Gtable).

```
    if (IsOnBorder == 1)
    {
        //prevent X and Y move into any triangle.
        if (InTriangle(X, Y, Index_in_Gtable))
            return 1;
    }
```

```
    firstCorner = i;
    lasti = -1;
```

```
    do{
        if (i == lasti )
            return 0;

        if ( ( (gx[v[n(i)]] - X)*(gy[v[p(i)]] - Y) - (gx[v[p(i)]] - X)*(gy[v[n(i)]] - Y) ) < 0 ) //X1*Y2 + X2*Y1 = cross product < 0 ==> clockwise ==> flip happens
```



```



return 1; //flip happens

lasti = i;

if(o[n(i)] != -1)
    i = n(o[n(i)]);
}
while (i != firstCorner);

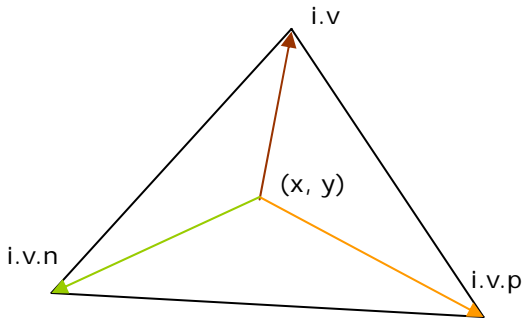
return 0;
}

```

As the figure shown, if the cross product of  &  < 0 flip happens.

## More on prevent overlap

To determine overlap or not, one need to know whether (X,Y) is located in a triangle or not, which can be compute as follow:



$$\begin{aligned}
 & (\vec{i.v.n} \times \vec{i.v}) * (\vec{i.v} \times \vec{i.v.p}) > 0 \\
 & \&\& \\
 & (\vec{i.v} \times \vec{i.v.p}) * (\vec{i.v.p} \times \vec{i.v.n}) > 0
 \end{aligned}$$