Search Area Path Generator

# Assumptions

* Search area polygons will contain internal angles of less than 180 degrees.

# Inputs

## Data

#### Raw Waypoints

Waypoints the plane must travel through.

#### Search Area

A polygon the plane must search.

#### Boundary

A polygon where the plane cannot pass through.

#### Starting Location

Coordinates of where the plane takes off.

#### Ending Location

Coordinates of where the plane lands.

## Parameters

#### Minimum Turning Radius

The minimum turning radius in metres.

#### Orientation Input

A custom axis of orientation if the user provides one.

#### Boundary Resolution

An inversely proportional factor to the step size that the path will adjust to avoid the boundary.

#### Boundary Tolerance

The minimum distance from the boundary the path must remain.

#### Curve Resolution

The number of waypoints per metre.

#### Paint Overlap

Minimum paint overlap required in terms of percentage. Must be less than 100%.

#### Sensor Size

The size of the camera sensor in mm. (Height, Width)

#### Focal Length

The focal length of the lens in mm.

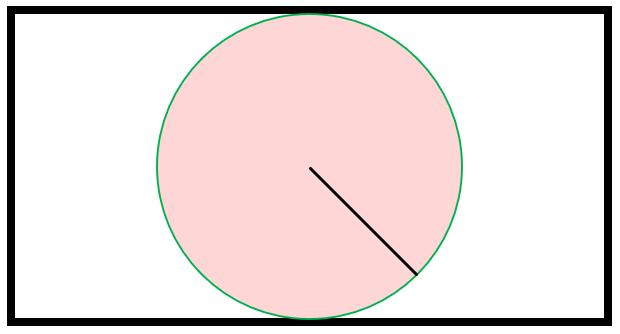
#### Max Flight Time

The maximum flight time of the plane.

# Pre-Calculations

#### Viewing Radius

Viewing radius is the radius the camera can see with reference to the ground.



If the black rectangle is the frame of the camera, the green circle defines the viewing radius.

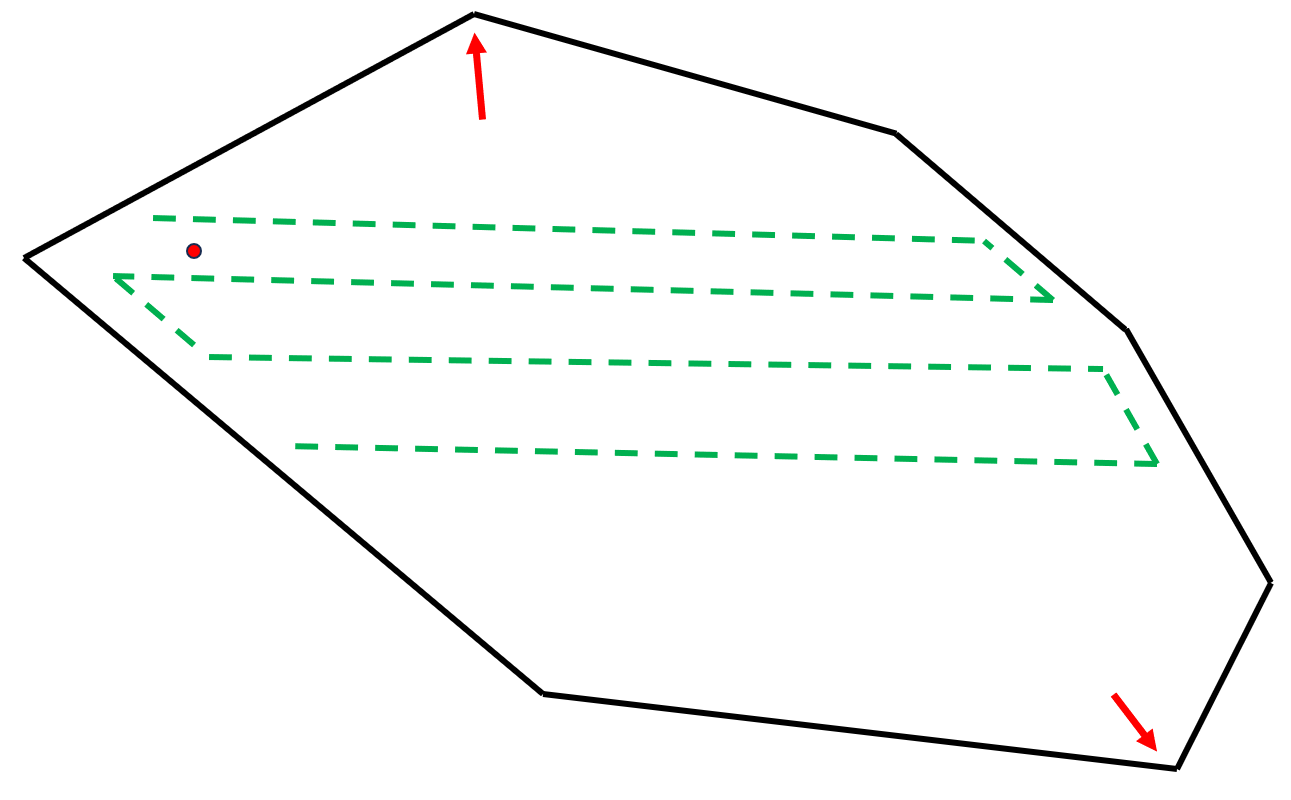
# Logic Structure

## Option 1 – User Defined Flight Path Orientation

The user inputs what flight path orientation they desire, and the script designs a flight path at that angle.

1. Find the starting and ending location of the flight path.

If the starting location is not within the search area, find the closest point along the search area boundary from the starting location. This point is (). If the starting location is within the search area, the starting location is ().

A drawing of a hexagon with green lines

Description automatically generated with low confidence

If the axis of rotation is , in the example above , then the path start will be at the furthest point in the direction of , shown by the red arrows. The direction, or , will be chosen based on which path start would be closest.

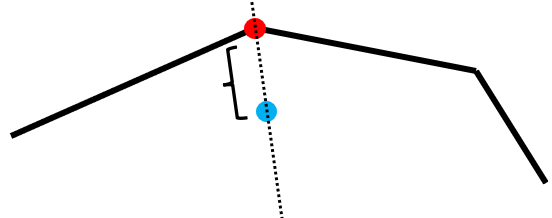
The coordinates of the path start will be on a vertex of the search area. Find the perpendicular vector () to the vector in the direction of the flight path orientation ().

Next, define each vertex as a vector () from the starting location () to the vertex ().

Then find the projection of each vertex vector () along the perpendicular vector (). Call this projection ().

Find the furthest vector in each direction () and pick the closest of the two as the path start and the furthest location as the path end. These are the two arrows in the example.

1. Offset the starting location from the search area boundary by the appropriate amount to properly “paint” the search area.



The point then needs to be moved away from the flight path boundary by metres. This variable is the distance required to most efficiently “paint” the search area whilst respecting the amount of paint overlap desired by the user.

Where is the radius of how much ground the camera can see at a given altitude and is a fraction which represents the percentage of the radius that the camera will overlap with previous images.

The direction of the path start point will have to have equal angles to its adjacent segments (black lines). Using the coordinates of centre vertex (red dot), and its adjacent vertices, the direction can be found.

Then using the distance () and the newfound angle above, the path starting point () can be found.

1. Pick a direction in either the direction of flight path orientation or of that. Travel in that direction until a search area boundary is within metres.

In terms of programming this, a point must be found that is in the direction of flight path orientation from the start point that is also metres from the search area boundary. A ray is cast at length from the path start in the direction of travel (). The intersection point of the ray and the search area polygon is found. The length is defined as the maximum distance between to vertices of the search area polygon.

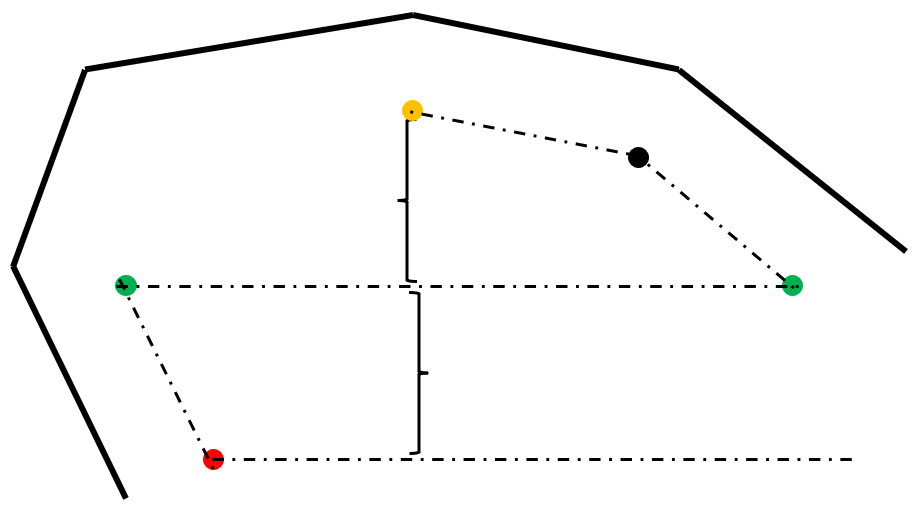
Then iterate over each segment of the polygon and find the point where the lines intersect (). For details, see [raycast intersections](#_Raycast_Intersection).

A picture containing line, diagram

Description automatically generated

Now that is found, must be found to ensure the path travelled by the plane (from to ) in this instance is parallel to the search area polygon. This is to be the case when the plane can’t travel in the flight path orientation without moving within metres of the search area polygon.

is defined as the point closest to that is metres from the parallel segment and metres from .



Make sure the wall is followed at a distance until metres are between each coloured layer as shown above.

1. Make smooth turns at sharp points.

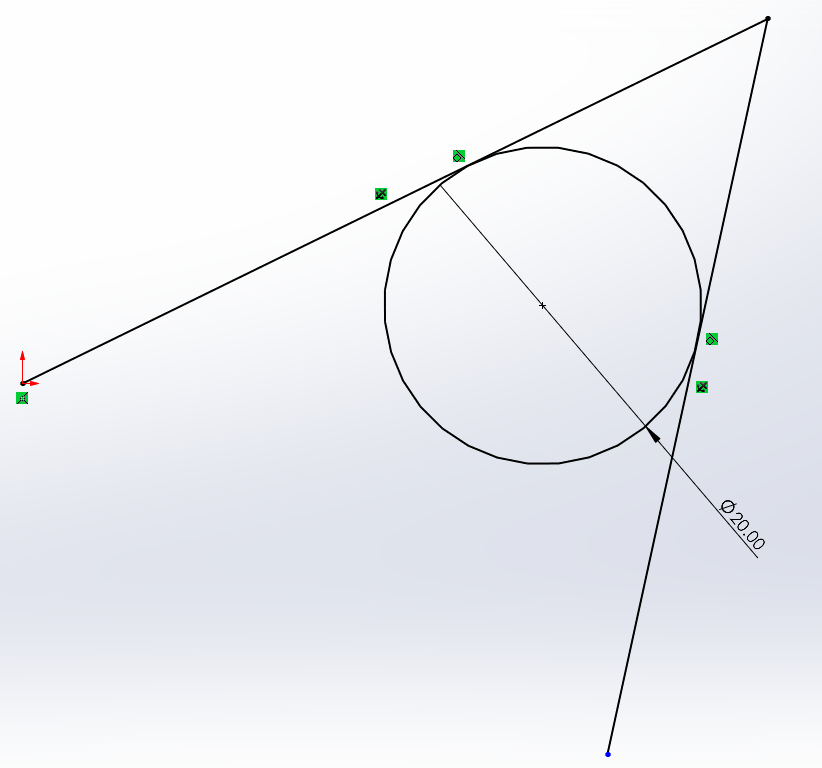
For cases where the turning radius (red) is larger than the viewing radius (green), a lightbulb turn will be executed as to not miss viewing any of the search area. See [executing a lightbulb turn](#_Executing_a_Lightbulb) for details.

A picture containing circle, diagram, illustration

Description automatically generatedA black and white outline of a keyhole

Description automatically generated with medium confidence

For cases where the turning radius is smaller than the viewing radius, a single circle can be used to turn. The internal angle of the turn will always be less than .

 A picture containing line, diagram, slope

Description automatically generated

Now the tangent points, and , are calculated as the vertices that create the internal angle are all known.

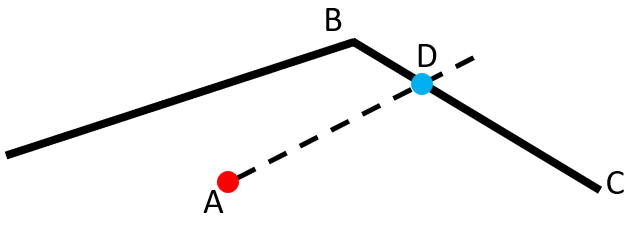
A drawing of a triangle

Description automatically generated with low confidence

# Function Details

## Raycast Intersection

This function casts a ray and returns the intersection point of the ray and another segment.

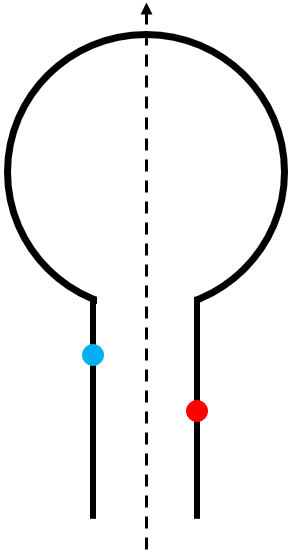


## Executing a Lightbulb Turn

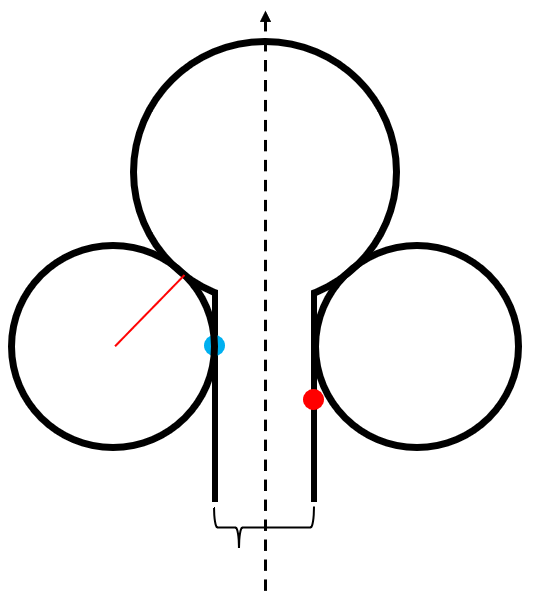
A lightbulb turn is usually required when the turning radius is too large for turn desired. This technique ensures that the start and end locations of the turn can be specified.

**Inputs:**

* Turn start coordinates (blue)
* Turn end coordinates (red)
* Axis of orientation (dashed line). Direction is important

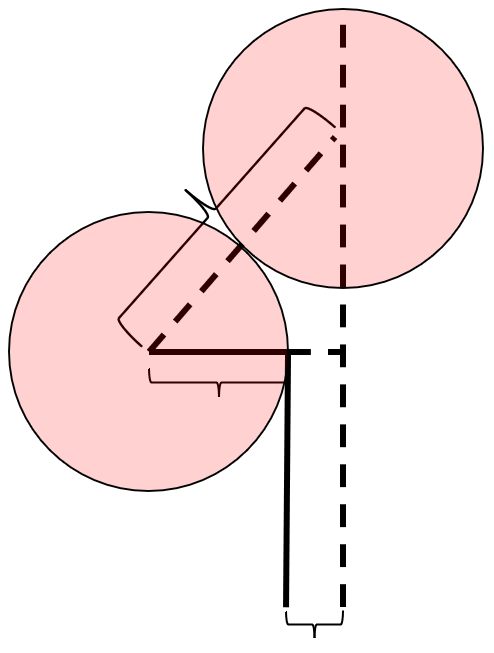


The function will generate two more circles to smooth out the turns. Each circle must have a radius greater than or equal to the minimum turn radius of the plane. The centre circle will move in the direction of axis of orientation to enable this. In this configuration, all circles can have a radius of the minimum turn radius.



### 

The location of the two side circles are perpendicular to the axis of orientation and at the minimum turn radius from the main path. The location of the middle circle can be found as one of the sides as a right-angled triangle.

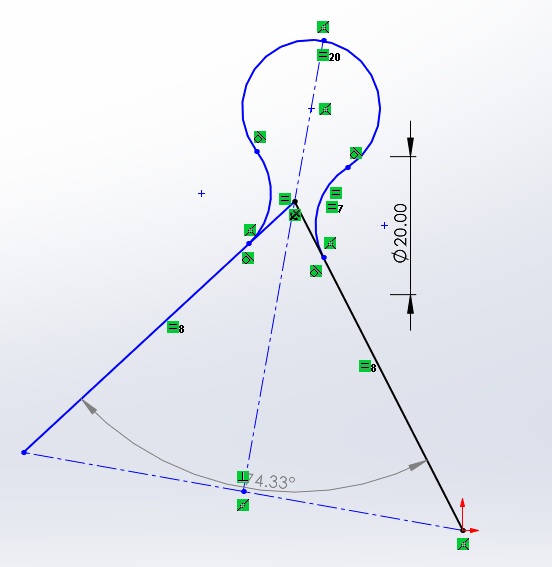


The coordinates of the middle circle are:

**If the blue coordinate () is ahead or equal to the red coordinate:**

**If the red coordinate () is ahead or equal to the blue coordinate:**

When the in and out of the turn aren’t parallel a slightly different technique must be used.



All circles are the same radius. Just use some geometry to figure out the circle centres.