Overview Package Class Tree Deprecated Index Help

Prev Class   **Next Class**   Frames   No Frames   All Classes
Summary: Nested | Field | Constr | Method   Detail: Field | Constr | Method

com._604robotics.robot2012.aiming

Class Aiming

java.lang.Object
     com._604robotics.robot2012.aiming.Aiming

```
public class Aiming
extends Object
```

Utility class for various aiming functions and such.

Author:

    Kevin Parker

Field Summary

Fields

Modifier and Type	Field and Description
static **Aiming**	**defaultAiming**

Constructor Summary

Constructors

Constructor and Description
Aiming()

Method Summary

Methods

Modifier and Type	Method and Description
PointAndAngle3d	**getAngleAndRelXYZOfTarget**(double x1, double y1, double x2, double y2, double x3, double y3, double x4, double y4) Get the angle from the targets, and the relative distances of the corners of the target as perceived by the camera.
double	**getAngleOfTarget**(double x1, double y1, double x2, double y2, double x3, double y3, double x4, double y4, double z) This function gets the direction the target is facing, relative to the camera.
Point3d	**getRelXYZOfTarget**(double x1, double y1, double w, double h) Remember that this requires the camera to be "perfectly" flat, and the targets to be "perfectly" vertical.
Point3d	**getRelXYZOfTarget**(**Target** t)

Methods inherited from class java.lang.**Object**

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

defaultAiming

```
public static final Aiming defaultAiming
```

Constructor Detail

Aiming

Method Detail

getRelXYZOfTarget

```
public Point3d getRelXYZOfTarget(double x1,
                                 double y1,
                                 double w,
                                 double h)
```

Remember that this requires the camera to be "perfectly" flat, and the targets to be "perfectly" vertical. A new function will probably need to be created for use on the robot. That, or we'll need to manipulate the points based on camera angle. The points are in the following pattern: +y ^ | 1 2 | | 3 4 +------> +x

Parameters:

x1 - x-value of the bottom left corner

y1 - y-value of the bottom left corner

w - width of the vision target

h - height of the vision target

Returns:

a Point3d holding the X, Y, and Z of the target, relative to the camera.

getRelXYZOfTarget

```
public Point3d getRelXYZOfTarget(Target t)
```

getAngleOfTarget

```
public double getAngleOfTarget(double x1,
                               double y1,
                               double x2,
                               double y2,
                               double x3,
                               double y3,
                               double x4,
                               double y4,
                               double z)
```

This function gets the direction the target is facing, relative to the camera. It is imperfect, and half-assumes a simple orthographic projection (which is not quite like real life). If it causes issues (which the accuracy of this function doesn't need to be very high), we can fix it later.

Parameters:

x1 - x-value of the bottom left corner

y1 - y-value of the bottom left corner

x2 -

y2 -

x3 -

y3 -

x4 -

y4 -

z -

Returns:

the resulting angle in radians.

getAngleAndRelXYZOfTarget

```
public PointAndAngle3d getAngleAndRelXYZOfTarget(double x1,
                                                 double y1,
                                                 double x2,
                                                 double y2,
                                                 double x3,
                                                 double y3,
```

```
                                    double x4,
                                    double y4)
```

Get the angle from the targets, and the relative distances of the corners of the target as perceived by the camera.

Parameters:

   x1 -

   y1 -

   x2 -

   y2 -

   x3 -

   y3 -

   x4 -

   y4 -

Returns: