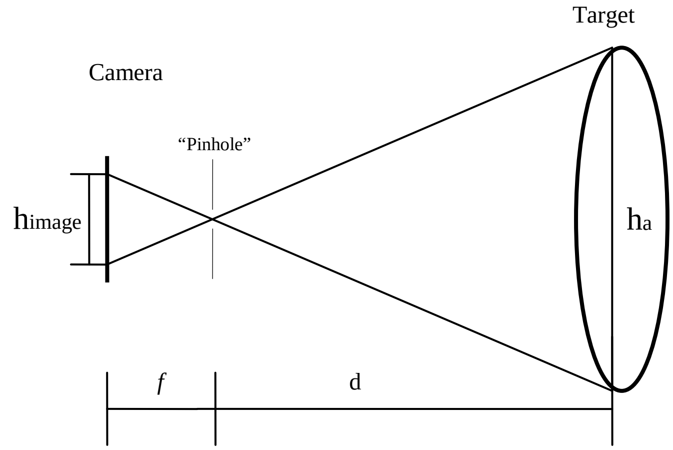
Determining Robot Position Relative to Vision Target by Analyzing Camera Image

Peter Stoeckl
26 January 2010
Revised 31 Jan 2010

Abstract: Using the FRC Camera in conjunction with the NI Vision software, the robot's position relative to a vision target can be determined. The Vision software can detect ellipses in the camera image and measure their major and minor diameters. The ellipse detection software can thus measure the apparent deformation of the target by the camera's changing angle relative to the target, which can be used to determine the robot's position.

First, it is necessary to examine the relation between the camera image and the target's dimensions.



Assuming the camera to be equivalent to a pinhole camera makes it easy to see that the height h_{image} of the target on the camera image and the effective focal length f of the camera are proportional to the target's apparent actual height (see below) h_a and the distance d from the camera to the target: $h_{image} \div f = h_a \div d$. The same applies to the respective widths: $w_{image} \div f = w_a \div d$.

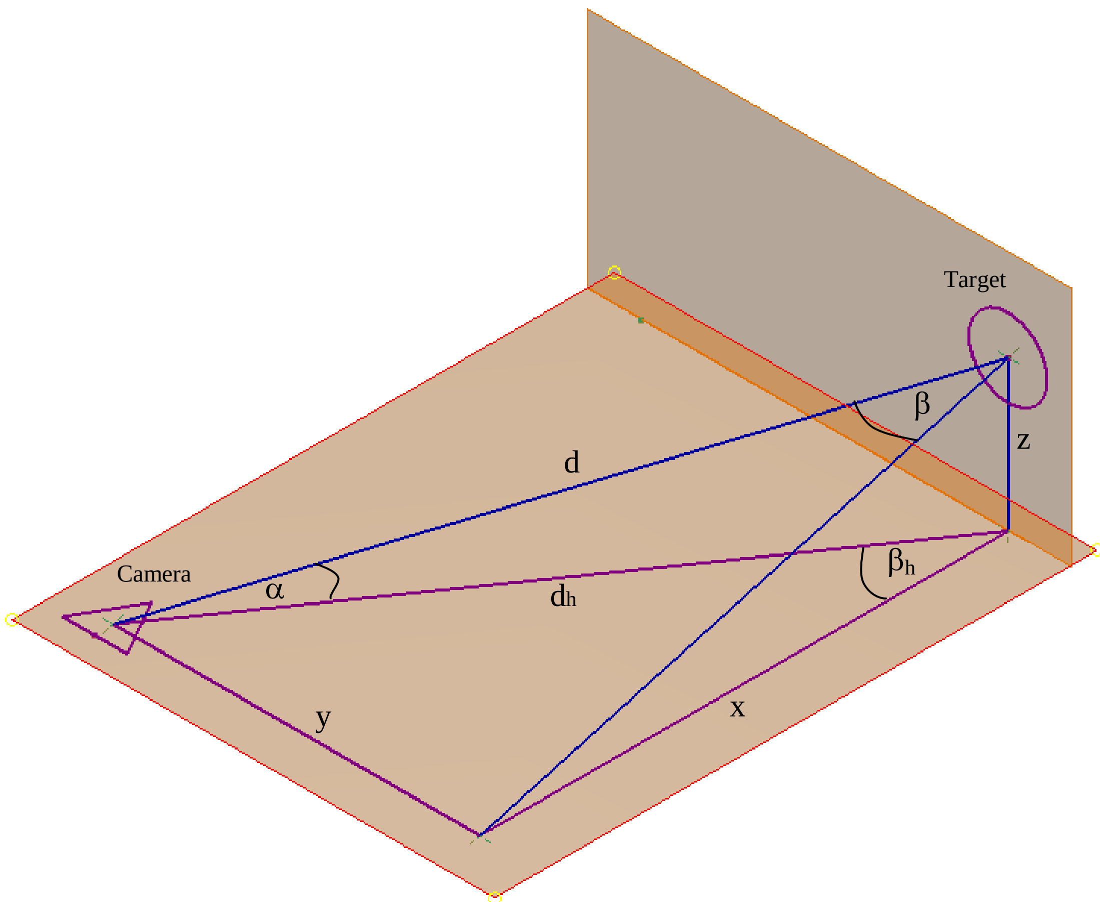
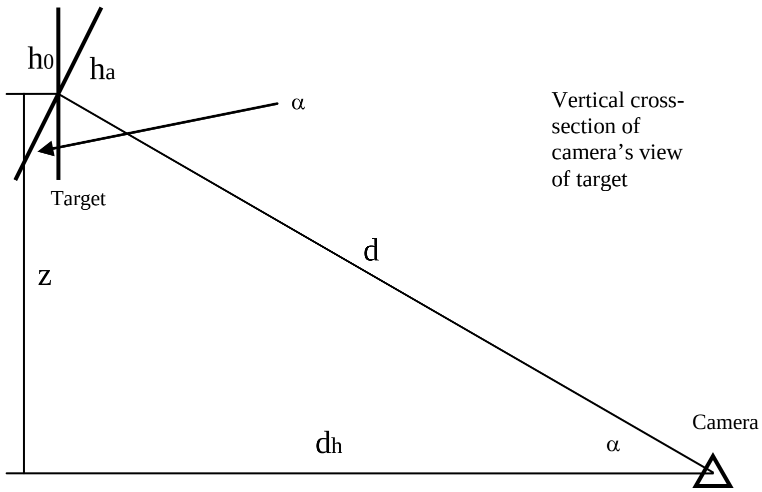
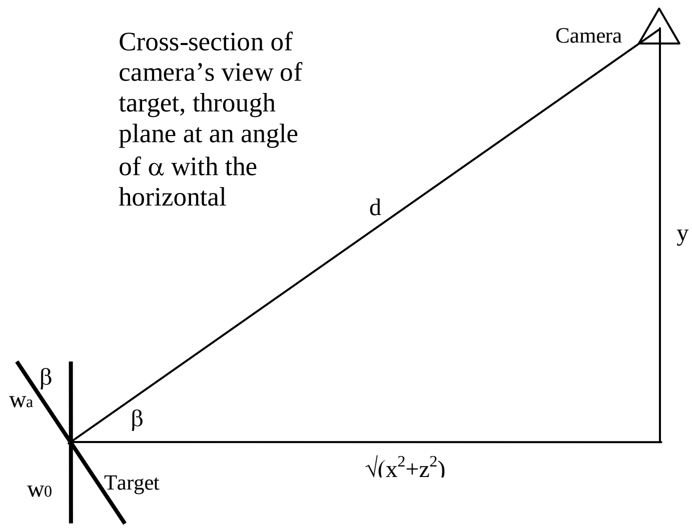


Diagram of the target as viewed by the camera



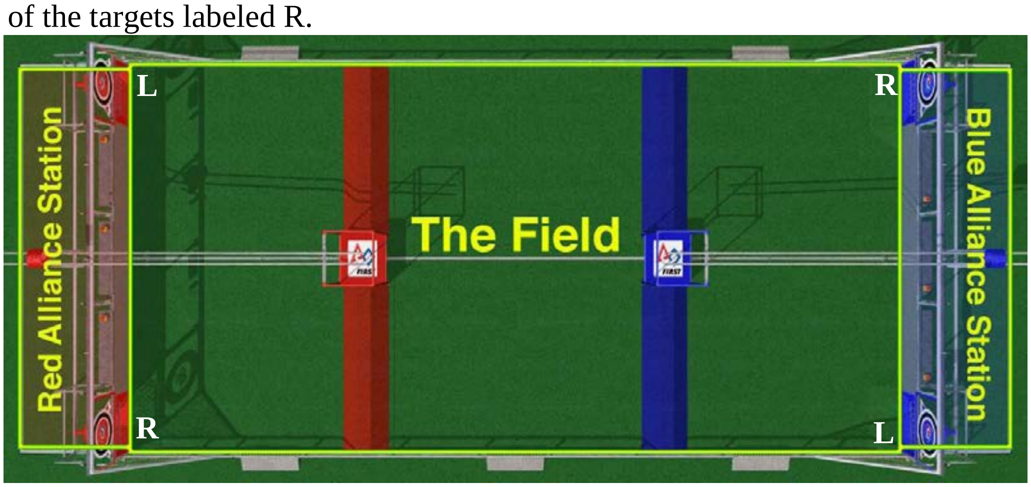
The vision target has a true diameter (thus height) h_0, which is known. The difference in elevation z between the camera and the center of the target is also known. The camera's (effective) focal length f can be determined by experimentation. At a constant horizontal distance d_h from the target, the height of the target on the image (h_{image}) remains constant. This value can be determined using the ellipse detection software. The camera is not detecting the true height of the target, however, but instead an apparent height h_a; the plane of this apparent height is at an angle of α to the vertical, which can be shown to be the same as the angle between the line from the camera to the center of the target (with length d) and the horizontal.

Then d is the distance from the camera to the target, and since $\sin \alpha = z \div d$, $d = z \div \sin \alpha$. Also, since the angle between the two h planes is α, $h_a = h_0 \cos \alpha$. Substituting these values into $h_{image} \div f = h_a \div d$ yields $h_{image} \div f = (h_0 \cos \alpha \sin \alpha) \div z$. $\cos \alpha \sin \alpha = ½ \sin 2\alpha$ (see, for example, www.clarku.edu/~djoyce/trig/identities.html), thus $h_{image} \div f = (h_0 \sin 2\alpha) \div 2z$. Solving for α yields $\alpha = (\mathrm{asin}\,((2zh_{image}) \div fh_0) \div 2$, in terms of variables which are all known. Then $\tan \alpha = z \div d_h$, therefore $d_h = z \div \tan \alpha$. Thus the horizontal straight-line distance, d_h, from the robot (with the camera) to the target can be determined.



From the first section, $w_{image} \div f = w_a \div d$. Solving for w_a: $w_a = dw_{image} \div f$. Also (by the same argument as in the second section) $w_a = w_0 \cos \beta$, w_0 being equal to h_0 since the target is circular. Substituting, $w_0 \cos \beta = dw_{image} \div f$, and then solving for β:
$\beta = \operatorname{acos} (dw_{image} \div fw_0)$. (Note that d equals $z \div \sin \alpha$—see previous section). This is, however, not in a horizontal position (see 3-D diagram); the true horizontal angle β_h is therefore different from β. The length y is equal to $d \sin \beta$, and $d_h = d \cos \alpha$; then $\sin \beta_h = y \div d_h = d \sin \beta \div d \cos \alpha = \sin \beta \div \cos \alpha$, and therefore $\beta_h = \operatorname{asin} (\sin \beta \div \cos \alpha)$. The angle of the robot (and camera) from the target has now been determined. In addition, it is now possible to locate the robot with cartesian coordinates relative to the target; the distance in one direction (y) has already been calculated; in the other direction, $x = d_h \cos \beta_h$.

Final Note: This algorithm does not give enough information to determine whether the robot is β_h degrees to the left or to the right of the target. However, if it is known which target on the Breakaway field is being observed, this can be determined to a reasonable degree of certainty. Since the targets are very close to the sides of the field, it is very unlikely that the camera will be on the side of the target nearer the side of the field; therefore the robot will most likely be to the left of the targets labeled L below and to the right of the targets labeled R.

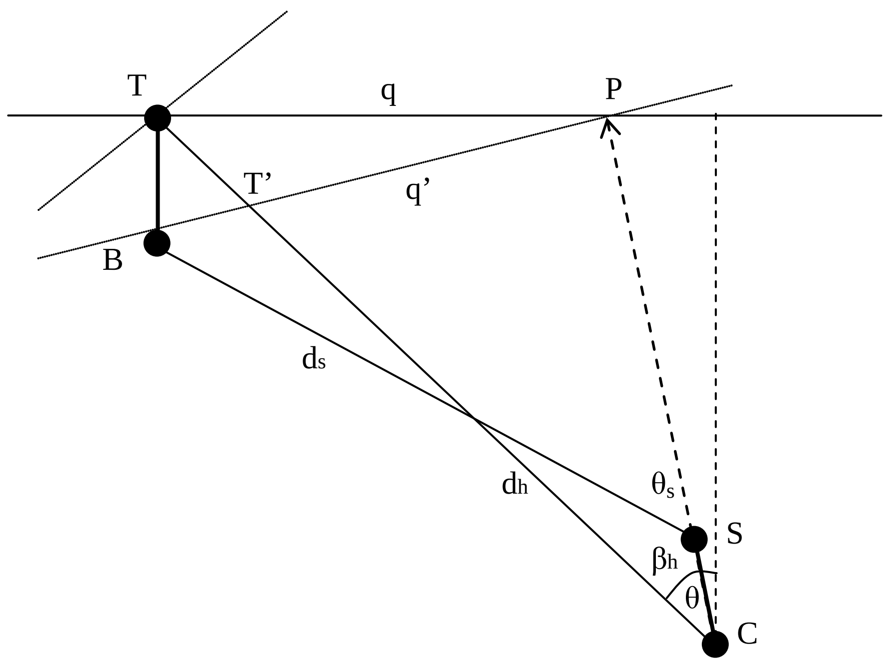


Addendum: 2012-Specific Additions

17 Feb 2012

The calculation of position (or angle+distance) occurs in mostly the same way, except that the target is now a rectangle, not an ellipse, so it has independent height and width values.

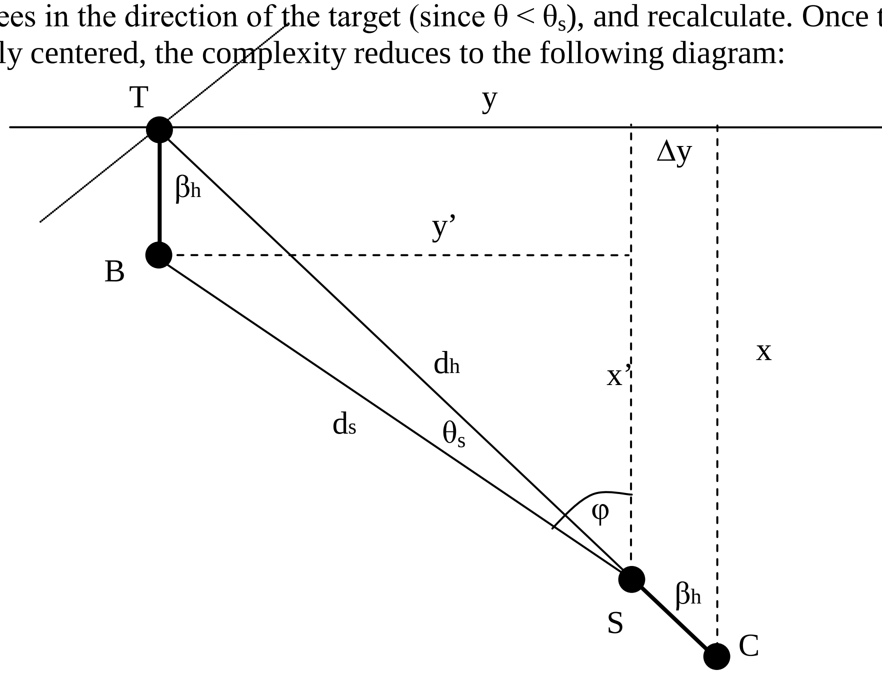
Using a camera on a turret:



In this diagram, the center of the target is at point T, the center of the basket attached to the target is point B, the center of the turret shooter is point S, and the camera is at point C. The angle β_h and distance d_h are calculated as before. The desired values, however, are d_s (distance from shooter to basket) and θ_s (change in angle between current turret position and pointing straight at the basket). Now the camera is pointed at a specific spot on the alliance wall (arrow), with an image plane (dotted line) perpendicular to the line of sight at the point it contacts the wall, and the target an image at point T'; however, for the target size position calculation, we assumed an image plane perpendicular to the line of sight CT at the target center. However, at large enough d_h and small enough angle θ between camera line of sight and line to target, the two planes can be assumed identical, as can the distance from camera to point P and to point T (aka d_h). Under the same assumptions, we can assume q and q' identical; then from PC ($\approx PT=d_h$) and PT' (=q') we can calculate $\theta=\mathrm{atan}(q'/d_h)$, which, since we are assuming θ to be adequately small, approximately equals q'/d_h. (q' can be calculated by multiplying the distance the center of the target appears from the center of the image, in pixels, by the scale factor at the distance d_h—this again contingent on the assumptions above.)

Now we were looking for θ_s, but we do not yet have enough information, and too many assumptions. Our strategy therefore is to request the turret to turn an additional θ

degrees in the direction of the target (since $\theta < \theta_s$), and recalculate. Once the target is finally centered, the complexity reduces to the following diagram:



This diagram looks equally complicated, but the geometry is easier. x and y can be calculated as in the original method. Now $\Delta y = SC * \sin(\beta_h)$, and $y' = y - \Delta y$ can be calculated, as can $x' = x - SC * \cos(\beta_h) - TB$. Then $\varphi = \text{atan}(y'/x')$ and the desired $\theta_s = \varphi - \beta_h$, while $d_s = \text{sqrt}(x'^2 + y'^2)$.

One more postscript: a new derivation of y that can save some calculation in the code:
$y = d * \sin(\beta) = z/\sin(\alpha) * \sin(\beta)$. Multiplying by $1 = \cos(\alpha)/\cos(\alpha)$, we get
$y = z/\sin(\alpha)*\cos(\alpha) * (\sin(\beta)/\cos(\alpha)) = z/\tan(\alpha) * \sin(\beta_h) = d_h * \sin(\beta_h)$!
(Actually, this can just be derived directly from the first, 3-D diagram… Oh well!

Connection to positioning: if the identity of the target is known (this requires some complicated comparison of target center positions…), so is its position in the field coordinate system. Knowing the x and y distances between the robot and the target (possibly requiring taking camera position on turret into account), we can then add these values to the target's known position, and thus derive the robot's position on the field.