6.2 Tracking Controller

Object tracking implies following a given object. However, in this approach and for the sake of program modularity, the tracker module does not need to be aware of what it is following nor of its properties. This approach was chosen because it ensures total independence of the tracker module. Its inputs are the current peripheral image coordinates $\vec{v} = \left[x_p, y_p \right]^T$ of the object to track and the desired coordinates, i.e. where the object is and where it ought to be. Usually, the desired coordinates, named Target $\hat{\vec{v}} = \left[\hat{x}_p, \hat{y}_p \right]^T$ are the image's center. However, that may not always be the case. In this case and because the foveal camera is positioned on top of the peripheral image (in which tracking occurs), $\hat{\vec{v}}$ should be placed above the image's center as well. This problem will be addressed further ahead (chapters **Error! Reference source not found.** and **Error! Reference source not found.**). For now, the important issue is of how to bring the recognized object located at current coordinates $\vec{v}$ to some predefined target coordinates $\hat{\vec{v}}$

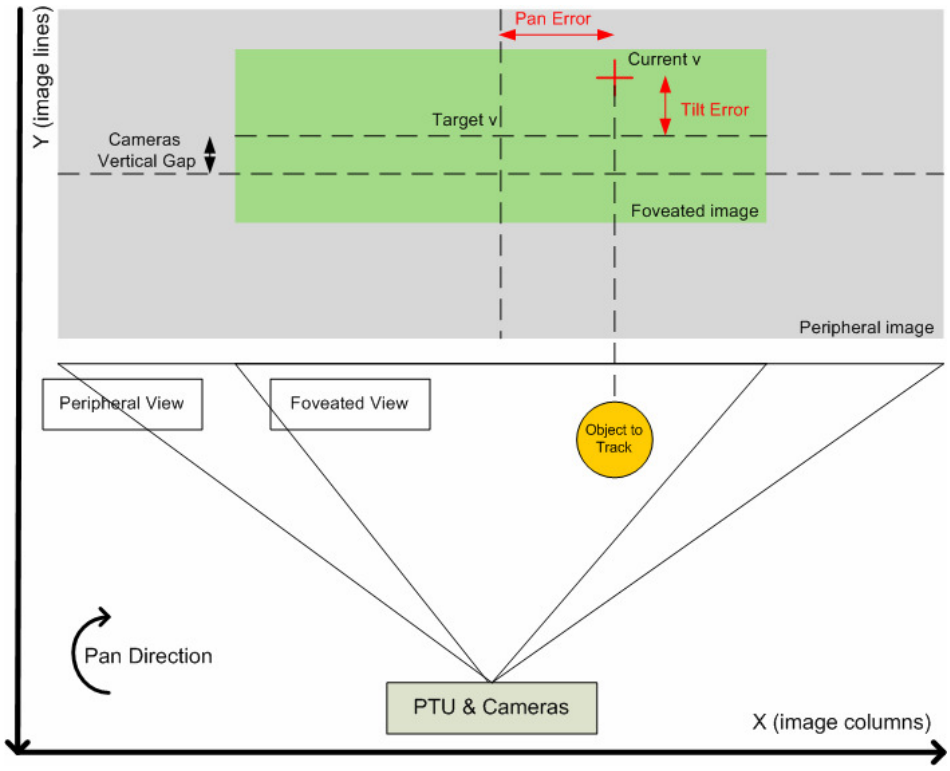

Figure 1. Pan & Tilt controller schematics and controller implementation.

Figure 1 illustrates a schematic of the perception unit. The pan error (P_{error}) is given by:

$$P_{error} = x_p - \hat{x}_p \tag{1}$$

While the tilt error (T_{error}) is defined as:

$$T_{error} = y_p - \hat{y}_p \tag{2}$$

Two independent controllers, one for each image axis, i.e. one for pan control and one for tilt control, are implemented. Nonetheless, both controllers are similar and therefore only the Pan controller will be addressed. Note that the philosophy of the whole implementation is not to use vision for exact metric measurements, but conversely, to use it for some kind of self calibrating control using only correlations between pixel distances. Therefore, there is no need to calculate the world coordinates that correspond to $\vec{v}$ nor to $\hat{\vec{v}}$. Without possessing this information, it is hard to specify the desired angles to pan/tilt based only on the object image coordinates. These angles have to be specified to the hardware unit. To solve this problem, fixed angle values are set. That is, if the

object is on the right side of $\hat{\vec{v}}$ ($T_{error} > 0$), the pan position's value ($P_{position}$) is a predetermined one that pans the cameras entirely to the right, while the opposite occurs when the object is on the left.

$$\begin{aligned} &if\,(P_{error} > 0) \Rightarrow P_{position} = 200^{o} \\ &else \Rightarrow P_{position} = -200^{o} \end{aligned} \tag{3}$$

The controller is actually a speed controller (in practice, position only reports the signal of the speed value). Pan speed is controlled based on a PID controller that accounts for P_{error} magnitude, its past history and future trends.

$$P_{speed} = K_p P^n_{error} + K_i \sum_{n=0}^{N} P^n_{error} + K_d \frac{P^n_{error} - P^{n-1}_{error}}{\Delta t} \tag{4}$$

Where K_p, K_i and K_d are the proportional, integral and derivative gains respectively, n is the iteration index, N the max amount of iterations to account for, and Δt corresponds to the time that has elapsed between iterations n and n-1. The K_p, K_i and K_d parameters were tuned by the method of empirical calibration. Pan acceleration may also be controlled by a PID controller but for now it is set as a constant high value with acceptable results so far.

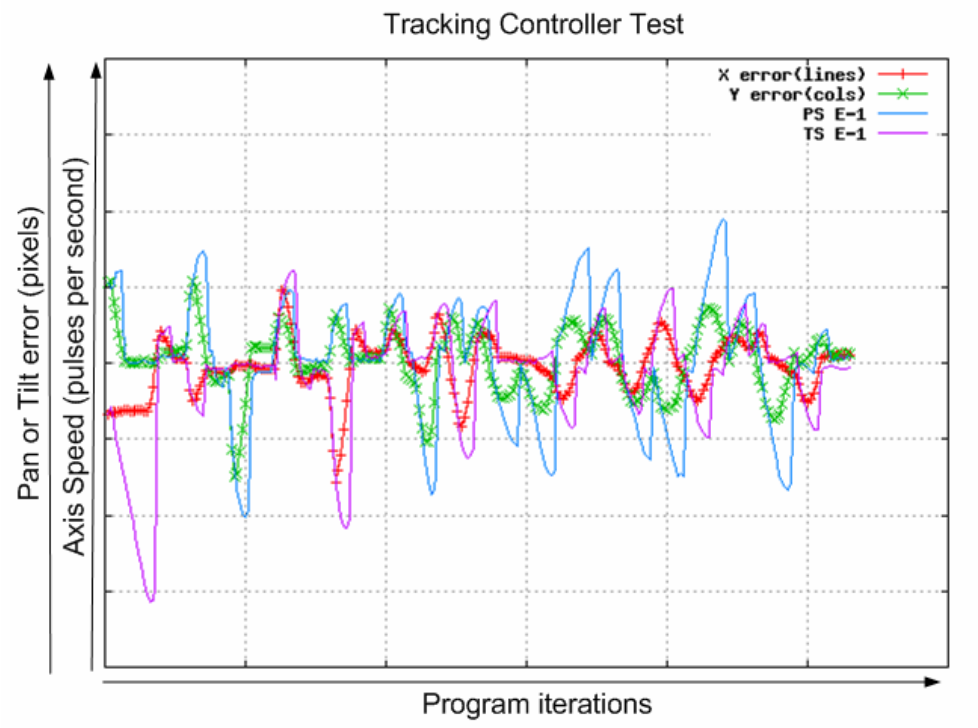

Figure 2. Tracking controller experiment.

Figure 2 shows how P_{speed} and T_{speed} vary when a P_{error} or T_{error} occurs. Independent axis control is noticeable since a T_{error} (x error in Figure 2) implies only that T_{speed} increases. Also, no overshooting is noticeable.