

Control Visual Basado en Posición Aplicado al Control de Vuelo del Drone Mavic 2 Pro

Proyecto de Control II

Marco Antonio Esquivel Basaldua

En este trabajo se presentan los resultados obtenidos al aplicar un control visual basado en posición, PBVC por sus siglas en inglés (Position Based Visual Servo), para el control de un dron Mavic 2 Pro en simulación. El simulador usado es *Webots* cuyos códigos son implementados en el lenguaje de programación Python. En las siguientes secciones de explica brevemente el PBVS, se dan las características del dron usado y se presentan imágenes y resultados al aplicar el control en simulación.

I. CONTROL VISUAL BASADO EN POSICIÓN (PBVS)

El control visual se puede considerar como el uso de técnicas de visión por computadora para el control del movimiento de un robot. En este tipo de control se acopla la percepción, mediante las imágenes captadas por una cámara, y la acción, de lo cual se encargan los actuadores propios del robot. Además se asume una configuración *eye-in-hand* en la que la cámara está puesta en el efecto final del robot.

El objetivo del control visual es minimizar un error $e(t)$

$$e(t) = s(m(t), a) - s^* \quad (1)$$

donde $m(t)$ es un conjunto de mediciones tomadas a partir de imágenes, $s(m(t), a)$ es un vector de características visuales medidas siendo a un conjunto de información adicional a la imagen (como características de la cámara o de la escena) y s^* corresponde al vector de características visuales deseadas. Típicamente nos referimos a este error de forma simplificada como

$$e(t) = s(t) - s^* \quad (2)$$

Un enfoque común es diseñar un controlador de velocidad el cual requiere la relación de las velocidades de las características visuales s y las velocidades de la cámara. Esta relación está dada por

$$\dot{s} = L_s \xi \quad (3)$$

donde $\xi = (v, \omega)$ corresponde al vector de velocidades de la cámara, siendo v las velocidades lineales instantáneas y ω las velocidades angulares, para esto se considera que la cámara puede moverse libremente en el espacio tridimensional. $L_s \in \mathbb{R}^{k \times 6}$ es la *matriz de interacción* o también llamada jacobiano de imagen la cual es determinada de acuerdo al tipo de control que se esté aplicando.

Derivando ahora la relación entre el cambio del error y las velocidades que controlan la cámara, se tiene

$$\dot{e} = L_e \xi \quad (4)$$

Asumiendo que el vector de características visuales deseadas, s^* , es constante, se llega a la relación

$$\dot{e}(t) = \dot{s}(t) = L_s \xi \quad (5)$$

por lo que $L_e = L_s$.

El PBVS hace uso de la información visual para estimar la pose de la cámara en un marco de referencia global el cual puede estar definido en el objeto visto por la cámara o en la pose deseada de la cámara. En este último esquema, el cual es usado para las simulaciones de este trabajo, el vector de características visuales se expresa como

$$s = ({}^c t_c, \theta u) \quad (6)$$

donde ${}^c t_c$ es el vector de tamaño 3 indicando la posición de la cámara con respecto a la posición deseada de la misma; θu es la representación de la rotación mediante un eje y un ángulo, este valor se puede obtener aplicando la fórmula de Rodrigues la cual se incluye en los anexos.

El vector de características deseadas indica el origen del marco de referencia global y por tanto es expresado como un vector de ceros de tamaño 6.

$$s^* = 0 \quad (7)$$

El error se define entonces como

$$e = s - s^* = s \quad (8)$$

En este caso la velocidad lineal en cada eje se calcula como la proyección del vector de velocidad lineal de entrada aplicado, ésta no se ve afectada por la velocidad angular. La matriz de interacción se define como

$$L_e = \begin{bmatrix} R & \mathbf{0} \\ \mathbf{0} & L_{\theta u} \end{bmatrix} \quad (9)$$

donde

$$L_{\theta u} = I_3 - \frac{\theta}{2} [u]_{\times} + \left(1 - \frac{\text{sinc}(\theta)}{\text{sinc}^2(\frac{\theta}{2})} \right) [u]_{\times}^2 \quad (10)$$

La ley de control es entonces

$$\begin{aligned} v &= -\lambda_v R^T c^* t_c \\ \omega &= -\lambda_\omega \theta u \end{aligned} \quad (11)$$

II. WEBOTS Y MAVIC 2 PRO

Webots es una aplicación de escritorio multiplataforma de código abierto usada para la simulación de robots. Provee de un ambiente para el modelado, programación y simulación de robots.

En esta aplicación se incluye el dron Mavic 2 Pro de DJI, figura 1, el cual consiste de un quadracóptero que incorpora una cámara de video. El control de este robot es llevado a cabo controlando las velocidades de rotación de cada una de sus cuatro hélices.

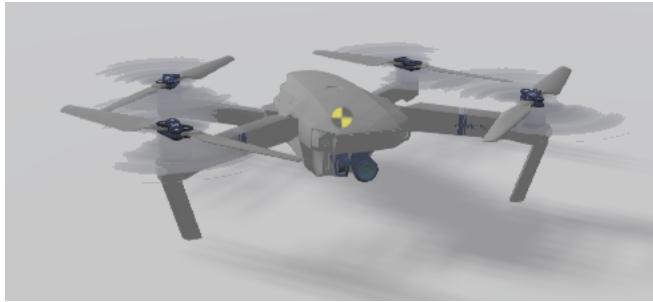


Figura 1. Mavic 2 Pro en Webots.

III. SIMULACIONES

Se implementó el controlador PBVS para el control de vuelo del dron mavic 2 pro en simulación. La idea es, con ayuda de la cámara del dron, a partir de una imagen objetivo, figura 2, mover al dron para que la imagen captada por el mismo se asemeje a la imagen objetivo. En la figura 3 se muestra la pose en la cual se tomó la imagen objetivo y, por lo tanto es la pose deseada del dron. De acuerdo con el controlador aplicado, el origen del sistema de referencia se encuentra en esta pose, a la cual se busca llevar al dron en las simulaciones con lo que el control se puede ver también como un problema de estabilización.

Se debe tomar en cuenta que los ejes no corresponden a los ejes trabajados en el controlador de acuerdo a la imagen, en lo siguiente el eje x corresponde a desplazarse hacia la derecha, el eje y a moverse hacia adelante y el eje z a moverse hacia arriba.

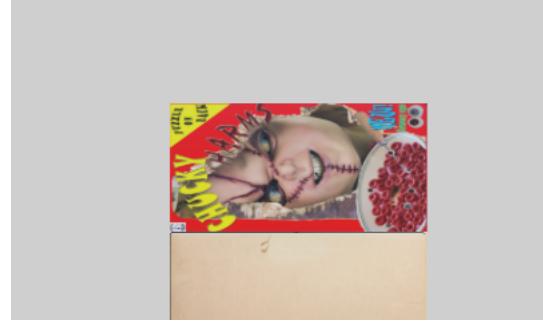


Figura 2. Imagen objetivo.

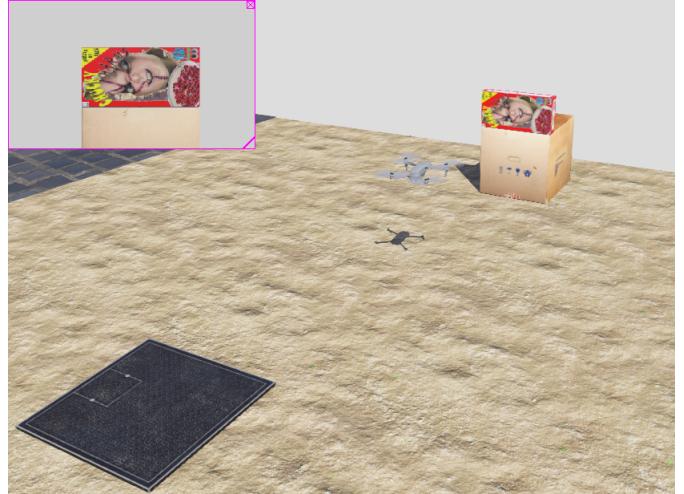


Figura 3. Pose del dron en la cual se tomó la imagen objetivo.

Hablar del PID y poner el diagrama La estimación de la pose actual del dron se lleva a cabo mediante la descomposición de la homografía calculada entre puntos clave emparejados entre la imagen objetivo y la imagen que se va obteniendo de la cámara a lo largo del vuelo del dron, un ejemplo de este emparejamiento se puede apreciar en la figura 4. En seguida es llevado a cabo el PBVS del cual se obtienen las velocidades de la cámara de las que se obtienen las necesarias para ser aplicadas al dron. Posterior a esto, se estima una pose deseada aplicando el método de Euler hacia la que se pide al dron que se mueva ya que su control está basado en un controlador PID de posición. Una vez aplicado el PID, se obtiene una nueva imagen de la que se obtiene la homografía y el proceso se vuelve a repetir. Este proceso se aprecia en el diagrama de bloques de la figura 5.

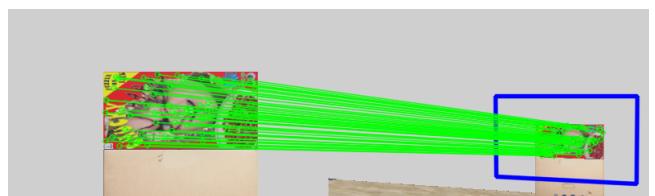


Figura 4. Puntos clave emparejados entre la imagen objetivo (izquierda) y la imagen actual tomada por el dron. El rectángulo en azul corresponde a los bordes de la imagen en la pose deseada.

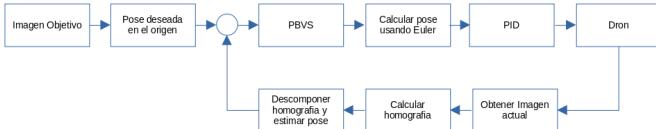


Figura 5. Diagrama de bloques del control aplicado en las simulaciones.

Se proponen cinco simulaciones en las que se aplica el control para distintas poses iniciales (cuatro para movimientos en cada grado de libertad por separado y uno más con movimientos generales) cuyos resultados se presentan a continuación. Cabe señalar que, aunque el mavic 2 pro cuenta con un motores para el control de la orientación de la cámara con respecto al dron, esto no se toma en cuenta y la orientación de la cámara solo se realiza mediante el control del ángulo ψ (yaw) en el vuelo del dron.

En estas simulaciones, las ejecuciones son detenidas (con el motivo de no extender el tiempo de simulación) cuando el dron se encuentra a una distancia menor a dos centímetros al origen estabilizando al robot en la última pose calculada. Lo que se espera es que el dron oscile al rededor del origen antes de estabilizarse. En el caso del ángulo ψ , éste se mide en el rango $[0, 2\pi]$ razón por la cual se observan saltos en las gráficas del ángulo de rotación.

A. Simulación 1

La pose inicial está dada por

$$t_{init} = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix}$$

$$\psi = 0$$

La pose inicial y la imagen tomada en esta pose se muestran en la imagen 4.

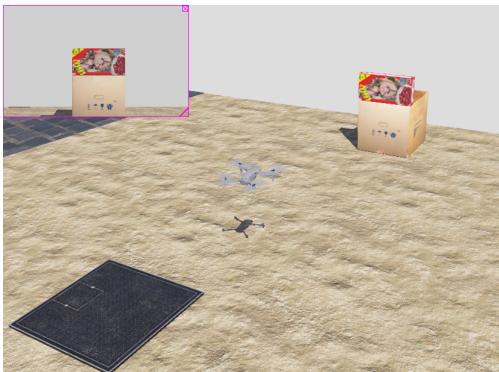


Figura 6. Pose inicial del dron en la simulación 1.

Al llevar a cabo el control se obtienen los siguientes resultados con respecto a la pose y a las velocidades.

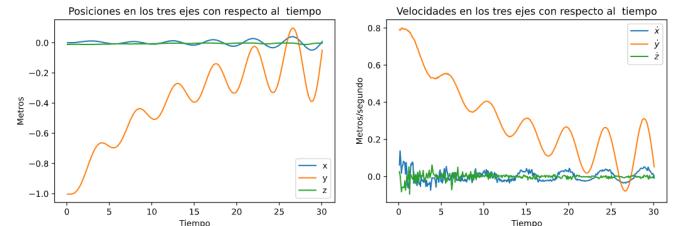


Figura 7. Evolución de la pose y las velocidades a lo largo del tiempo.

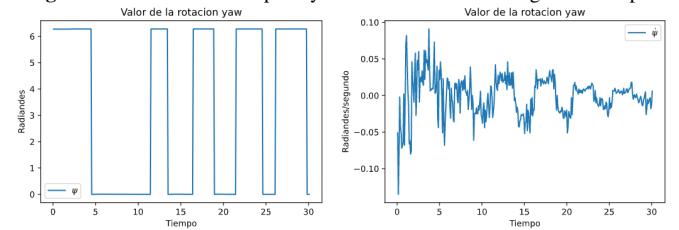


Figura 8. Evolución del ángulo de orientación y la velocidad de rotación a lo largo del tiempo.

B. Simulación 2

La pose inicial está dada por

$$t_{init} = \begin{bmatrix} 0.4 \\ 0 \\ 0 \end{bmatrix}$$

$$\psi = 0$$

La pose inicial y la imagen tomada en esta pose se muestran en la imagen 7.



Figura 9. Pose inicial del dron en la simulación 2.

Al llevar a cabo el control se obtienen los siguientes resultados con respecto a la pose y a las velocidades.

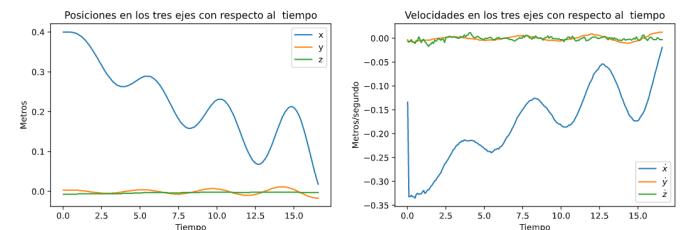


Figura 10. Evolución de la pose y las velocidades a lo largo del tiempo.

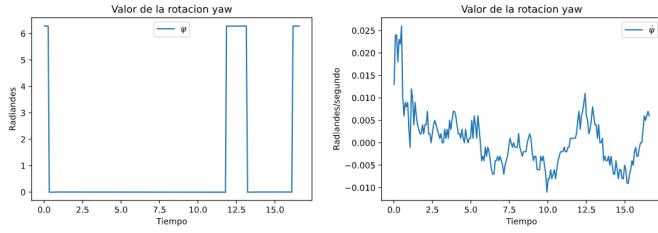


Figura 11. Evolución del ángulo de orientación y la velocidad de rotación a lo largo del tiempo.

C. Simulación 3

La pose inicial está dada por

$$t_{init} = \begin{bmatrix} 0 \\ 0 \\ 0.3 \end{bmatrix}$$

$$\psi = 0$$

La pose inicial y la imagen tomada en esta pose se muestran en la imagen 10.

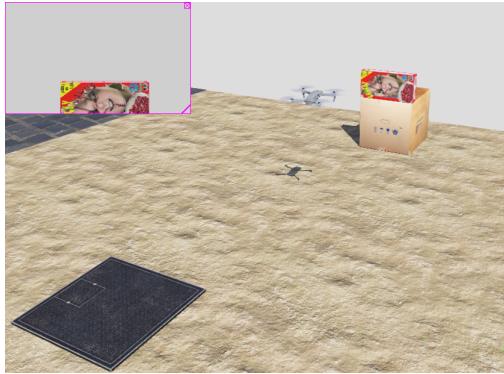


Figura 12. Pose inicial del dron en la simulación 2.

Al llevar a cabo el control se obtienen los siguientes resultados con respecto a la pose y a las velocidades.

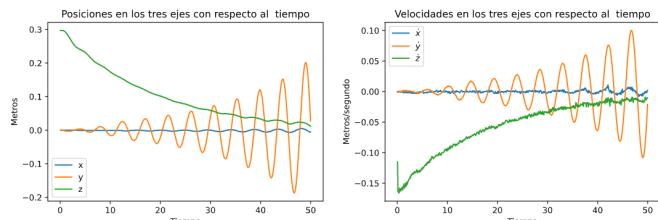


Figura 13. Evolución de la pose y las velocidades a lo largo del tiempo.

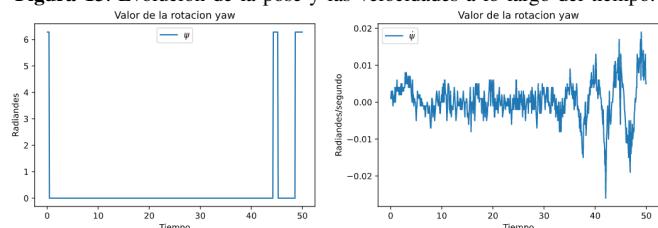


Figura 14. Evolución del ángulo de orientación y la velocidad de rotación a lo largo del tiempo.

D. Simulación 4

La pose inicial está dada por

$$t_{init} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\psi = 0.3$$

La pose inicial y la imagen tomada en esta pose se muestran en la imagen 13.

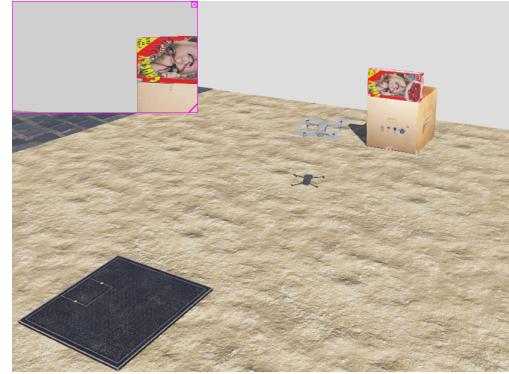


Figura 15. Pose inicial del dron en la simulación 2.

Al llevar a cabo el control se obtienen los siguientes resultados con respecto a la pose y a las velocidades.

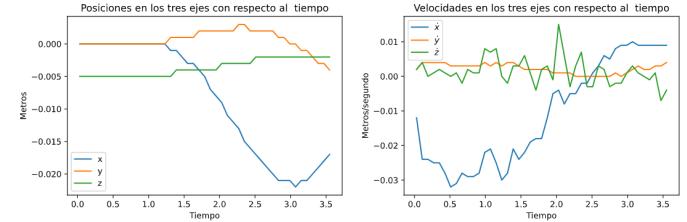


Figura 16. Evolución de la pose y las velocidades a lo largo del tiempo.

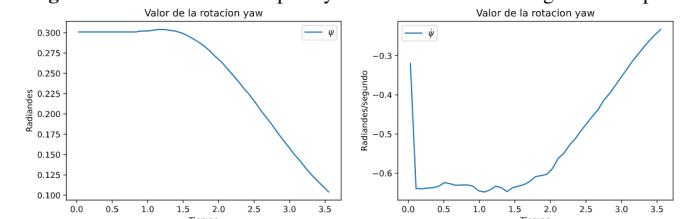


Figura 17. Evolución del ángulo de orientación y la velocidad de rotación a lo largo del tiempo.

E. Simulación 5

La pose inicial está dada por

$$t_{init} = \begin{bmatrix} -0.4 \\ -2 \\ 0.3 \end{bmatrix}$$

$$\psi = 0.3$$

La pose inicial y la imagen tomada en esta pose se muestran en la imagen 16.

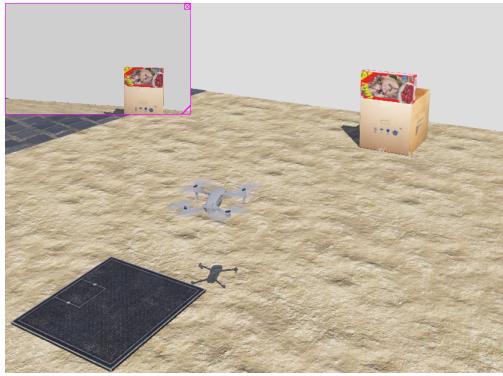


Figura 18. Pose inicial del dron en la simulación 2.

Al llevar a cabo el control se obtienen los siguientes resultados con respecto a la pose y a las velocidades.

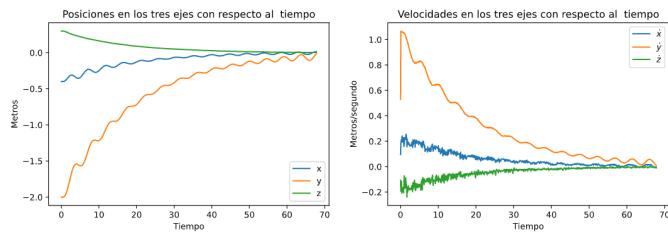


Figura 19. Evolución de la pose y las velocidades a lo largo del tiempo.

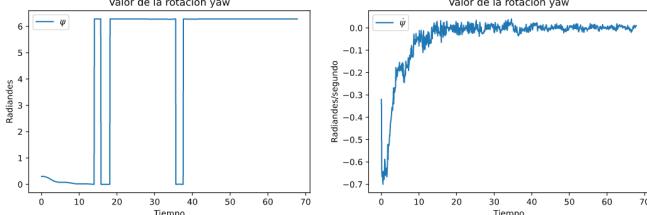


Figura 20. Evolución del ángulo de orientación y la velocidad de rotación a lo largo del tiempo.

IV. CONCLUSIONES

Al aplicar el PBCS se logra mover al dron a la pose deseada usando únicamente información visual dada por la cámara del dron. El dron presenta un comportamiento oscilante en su movimiento debido a que las velocidades calculadas por el controlador son usadas para estimar una pose deseada, aplicando el método de Euler, hacia la cual debe moverse el dron usando un controlador PID. En los casos en que pareciera que el dron diverge al llegar a la pose final, se esperaría que al llegar a este punto el dron oscile al rededor del origen hasta la estabilización. Una alternativa es detener el control y estabilizar el dron en la última posición calculada cuando las imágenes son lo bastante parecidas (este valor puede calcularse como la norma de las características visuales deseadas y actuales). Una alternativa más es mapear directamente las velocidades obtenidas del control a velocidades del dron sin la necesidad de estimar posiciones hacia las que se debe mover el dron.

La simulación que presenta mejores resultados es el caso en que se piden movimientos generales a la cámara (ninguno de los grados de libertad es puesto a cero al inicio de la simulación)

ya que se observa una convergencia al origen de cada uno de los grados de libertad.

V. ANEXOS

A. Fórmula de Rodrigues

Dado un eje de rotación representado por el vector $\mathbf{u} = [u_x, u_y, u_z]^\top$, la rotación se puede representar por

$$R(\mathbf{u}, \theta) = I + \sin(\theta)[\mathbf{u}]_\times + (1 - \cos(\theta))[\mathbf{u}]_\times^2 \quad (12)$$

Así pues, dada la matriz de rotación $R \in SO(3)$, la representación $\theta\mathbf{u}$ de dicha rotación se puede encontrar como

$$\theta = \arccos\left(\frac{1}{2}(r_{11} + r_{22} + r_{33} - 1)\right) \quad (13)$$

$$\theta\mathbf{u} = \frac{1}{2\sin(\theta)} \begin{bmatrix} r_{32} - r_{23} \\ r_{31} - r_{13} \\ r_{21} - r_{12} \end{bmatrix} \quad (14)$$