

# NAVIGATION SEMI-AUTOMATIQUE DE DRONE

## Equipe : MARS

# SOMMAIRE

## **Du Joystick à la Commande**

1.1 Joy\_teleop

1.2 Command

1.3 Behaviors

## **Conduite manuelle du Drone**

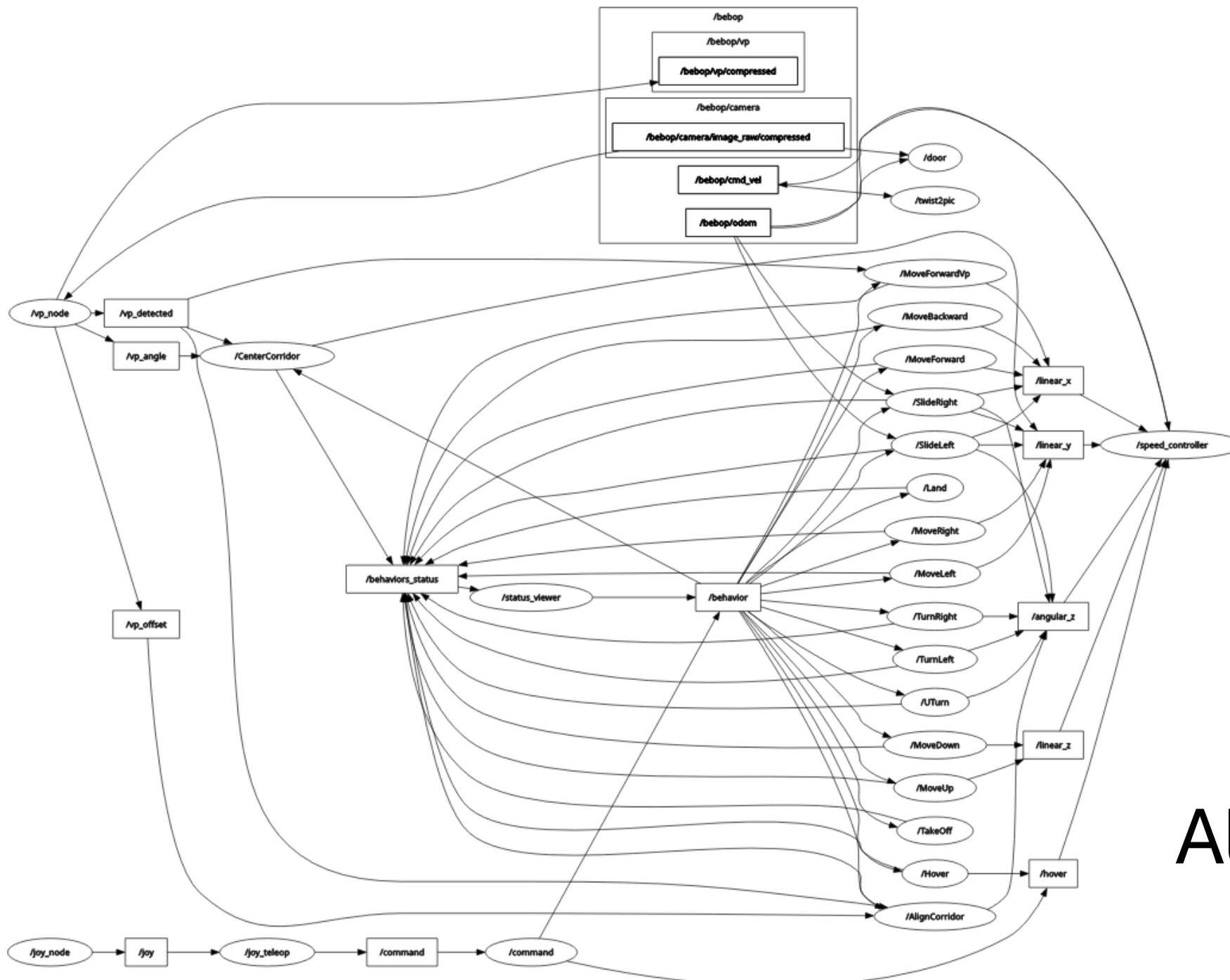
2.1 PID

2.2 UTurn

## **Conduite semi-autonome**

3.1 Vanishing point detection

3.2 Optical flow detection

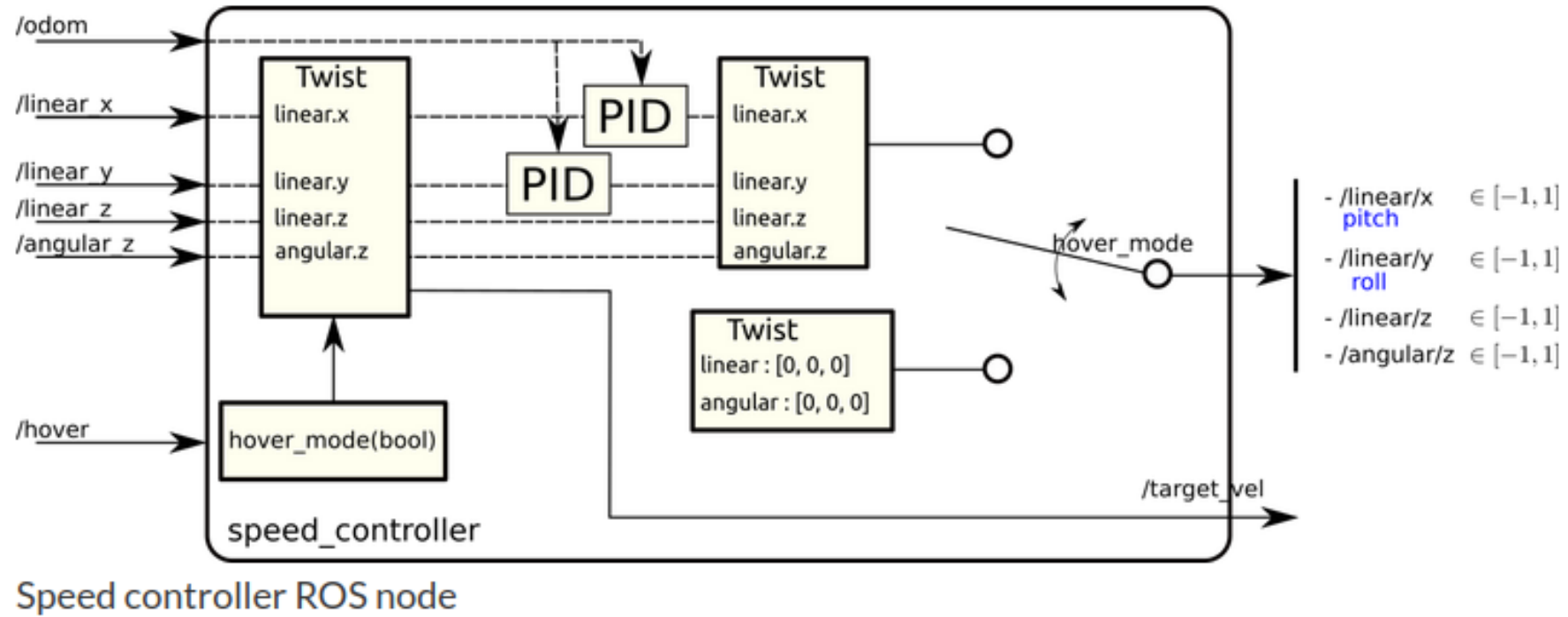


All Nodes and  
Topics



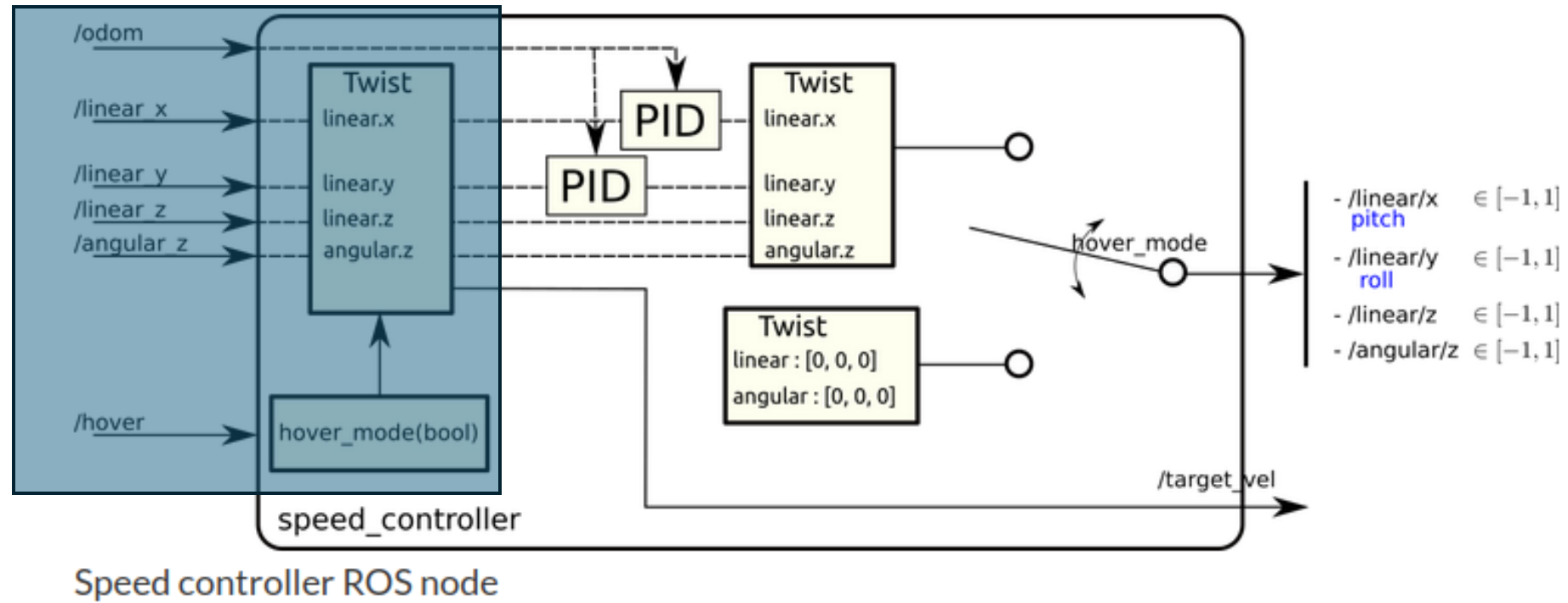
# NŒUD SPEED\_CONTROLLER

-> STRUCTURE GLOBALE DU NŒUD



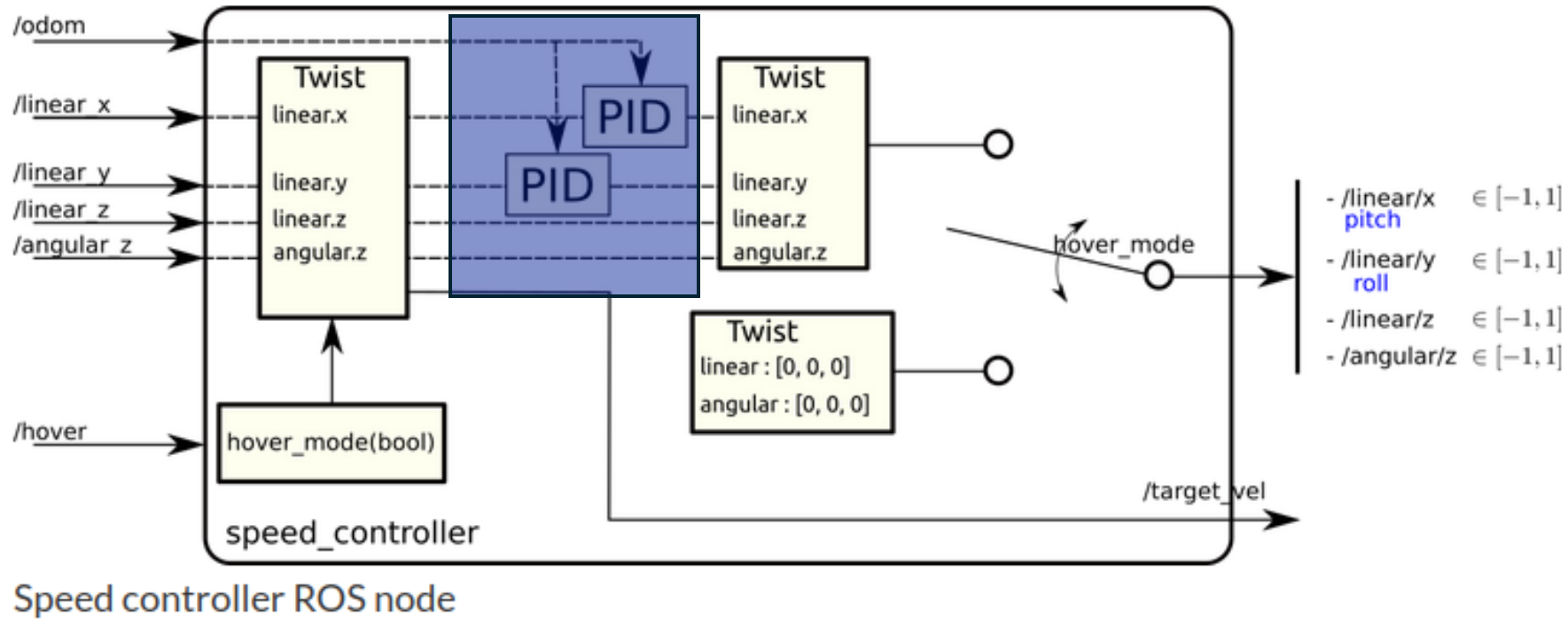
# NŒUD SPEED\_CONTROLLER

-> STRUCTURE GLOBALE DU NŒUD : ENTREES



# NŒUD SPEED\_CONTROLLER

-> STRUCTURE GLOBALE DU NŒUD : ETUDE DU PID



## NŒUD SPEED\_CONTROLLER

### -> Corrections des gains du PID

- ➔ D'abord sans correction, le drone volait correctement
- ➔ Puis on trouve expérimentalement qu'un gain de  $K_p = 1$  convient mieux



# NOEUD SPEED\_CONTROLLER

-> Corrections des gains du PID à l'aide de Ziegler-Nichols

-> Augmentation de  $K_p$  jusqu'à la valeur de  $K_u$  pour laquelle il y a oscillations

-> Calcul de  $T_u$  lorsqu'il y a oscillations



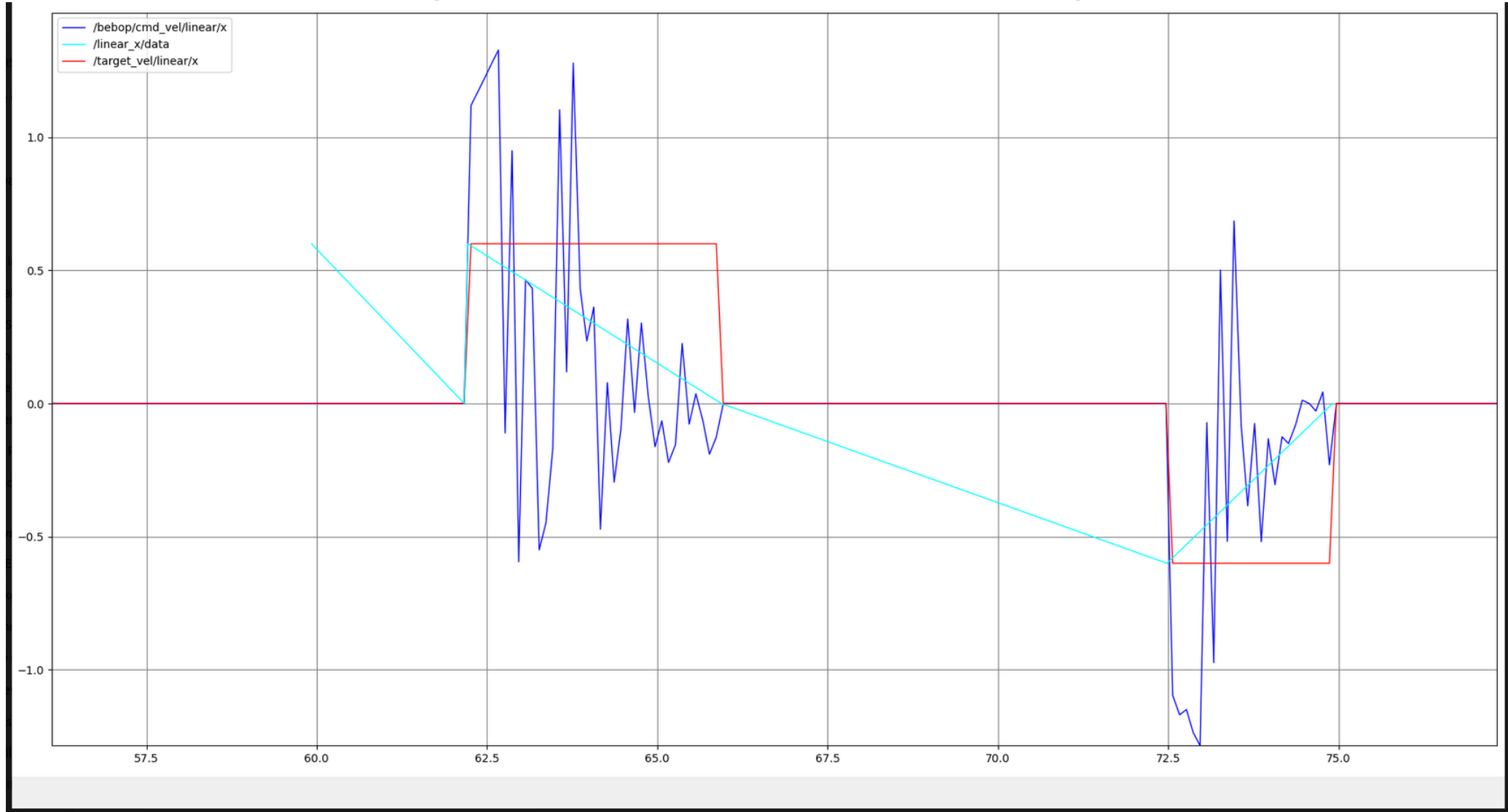
-> Valeurs de  $K_i, K_d, K_p$  avec les  $K_u$  et  $T_u$  selon le tableau suivant :

For twistx PID:

- $K_p$ : Increase by 10-20%. New  $K_p = 0.66 \times 1.4 \approx 0.92$
- $K_d$ : Increase by 10-15%. New  $K_d = 0.125 \times 0.92 \times 2.0 \approx 0.23$
- $K_i$ : Decrease by 20%. New  $K_i = 2 \times 0.92 / 2.0 \approx 0.92$

# NŒUD SPEED\_CONTROLLER

-> Correction des gains du PID à l'aide de rqt\_graph



## NŒUD SPEED\_CONTROLLER

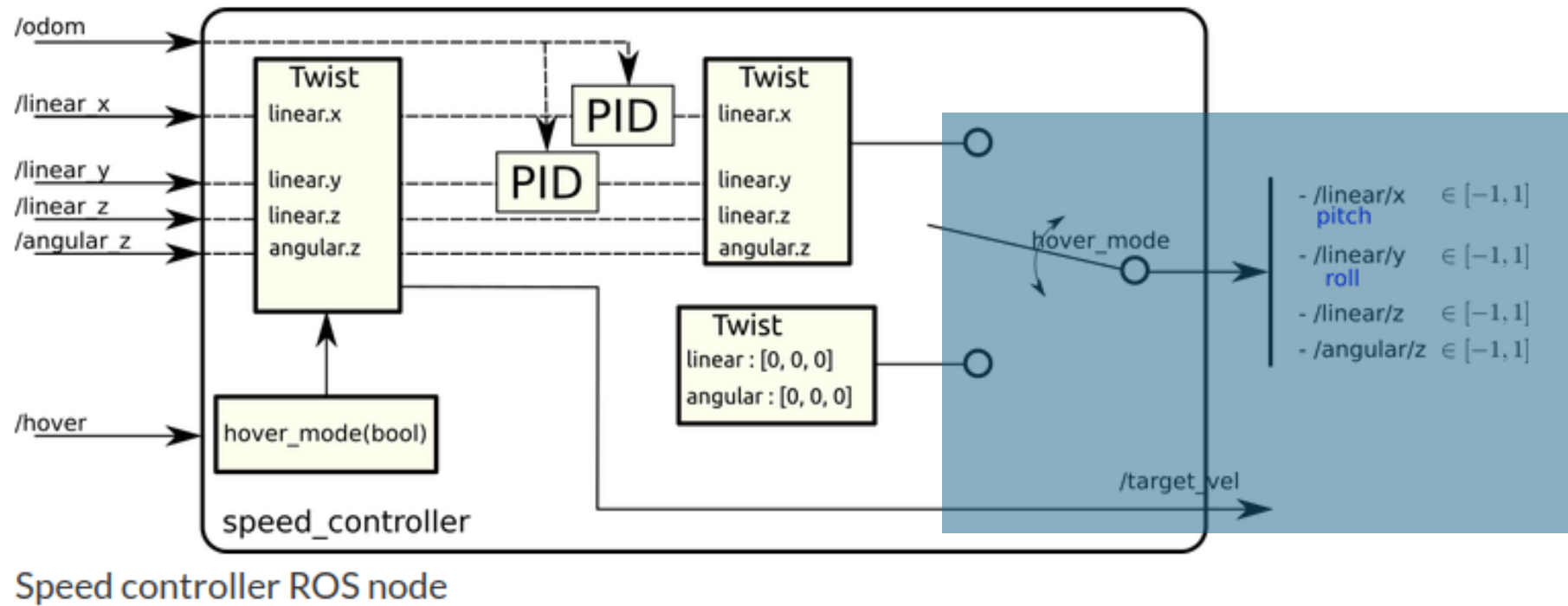
### Calibration du correcteur

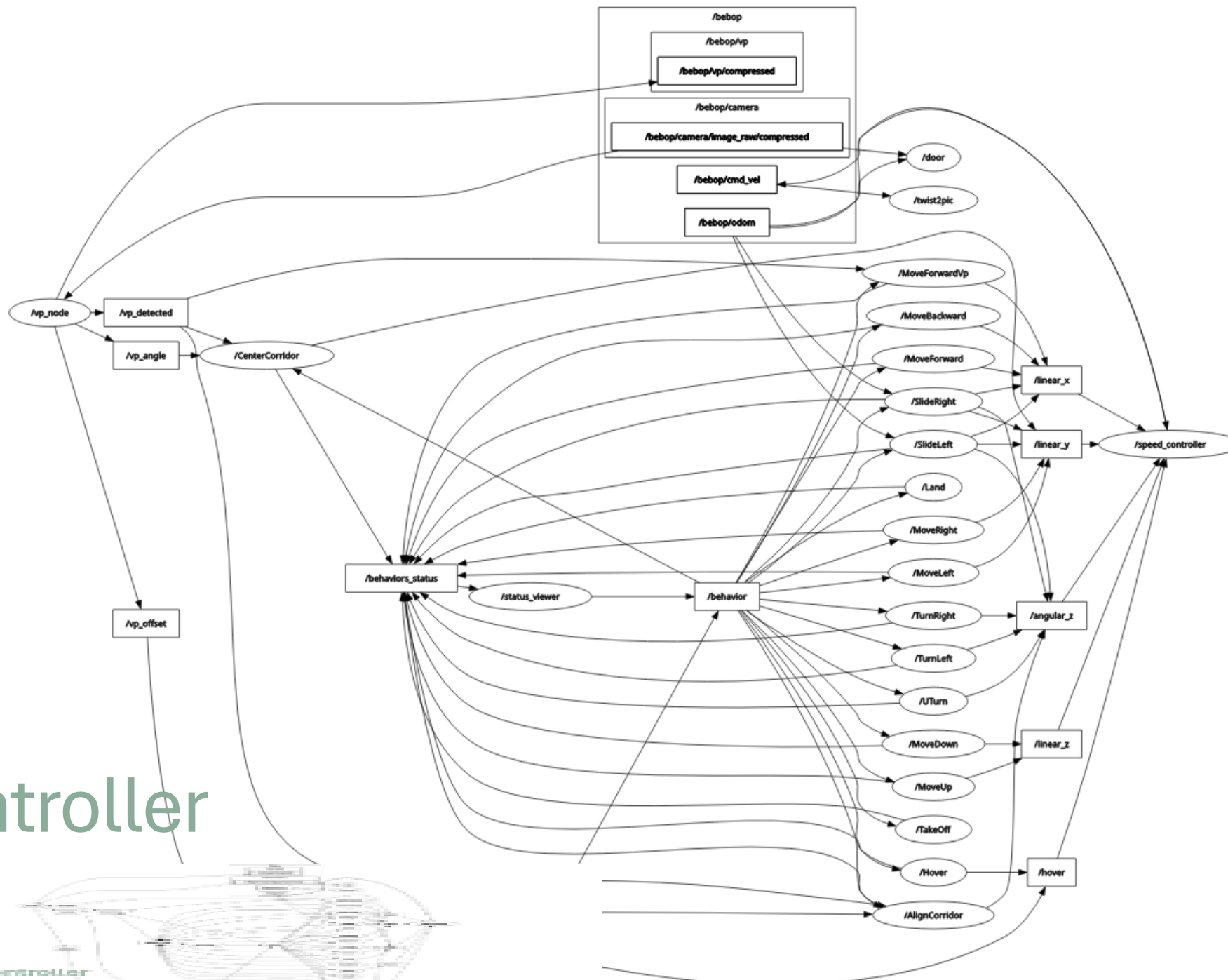
Au final on trouve ces valeurs expérimentalement

Valeurs	K <sub>p</sub>	K <sub>i</sub>	K <sub>d</sub>
Correcteur x	0.92	0.23	0.92
Correcteur y	0.92	0.23	0.92

# NOEUD SPEED\_CONTROLLER

## -> STRUCTURE GLOBALE DU NOEUD : SORTIES





Controller





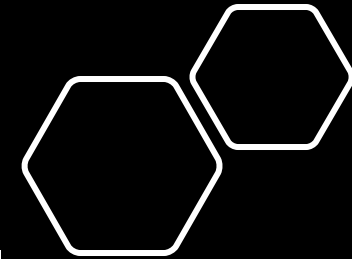
Foward/Backward  
Left/Right

TakeOff et Land

Hover

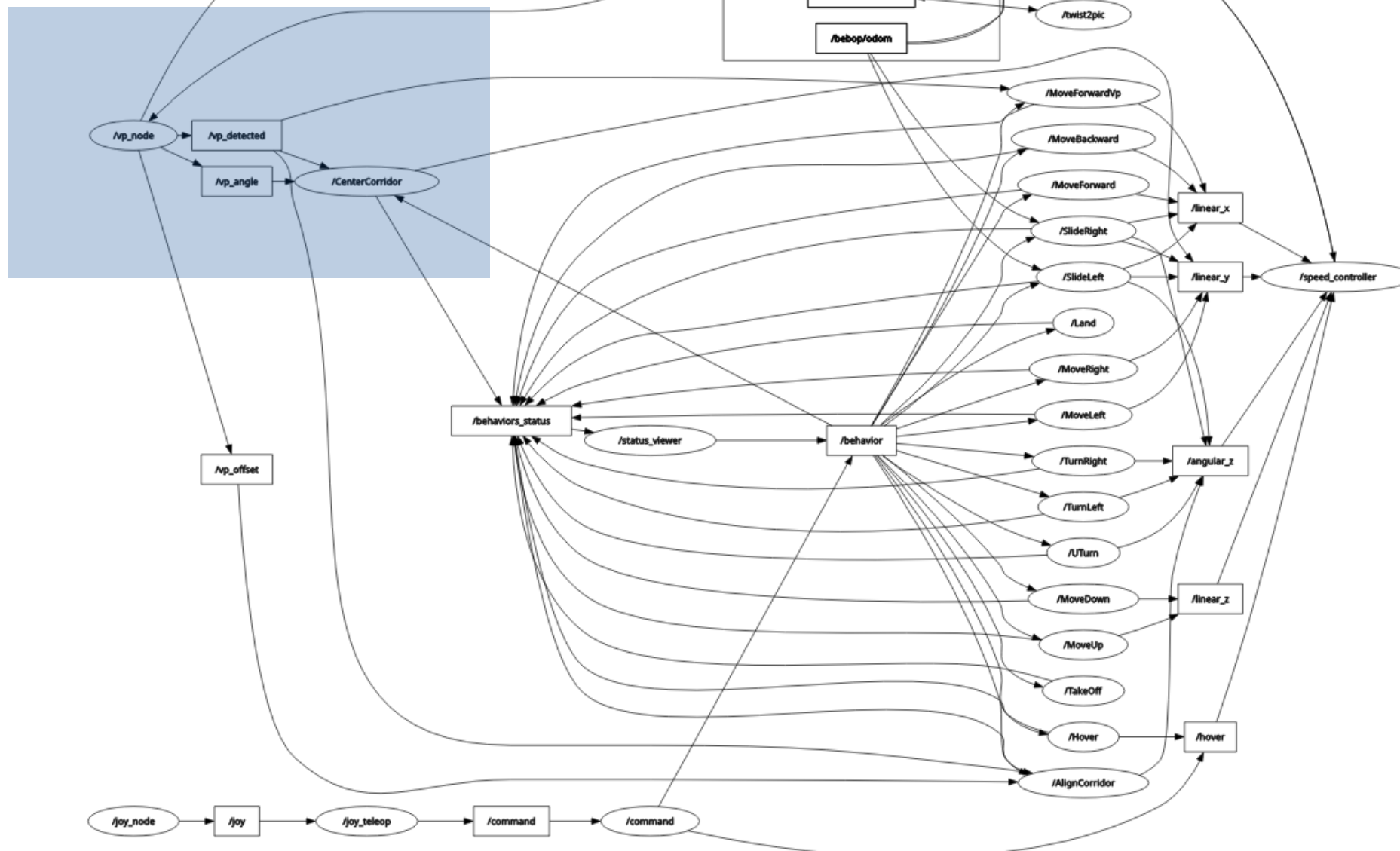
MoveUp et  
MoveDown

Rotation



Temps (s)	Action
0	UTurn
4.5	MoveForwardVp
5	AlignCorridor
5.5	CenterCorridor

# Vanishing point





# LA RECETTE DU VANISHING POINT

---

1 ligne de LSD from CV2

---

4 lignes de filtrage

---

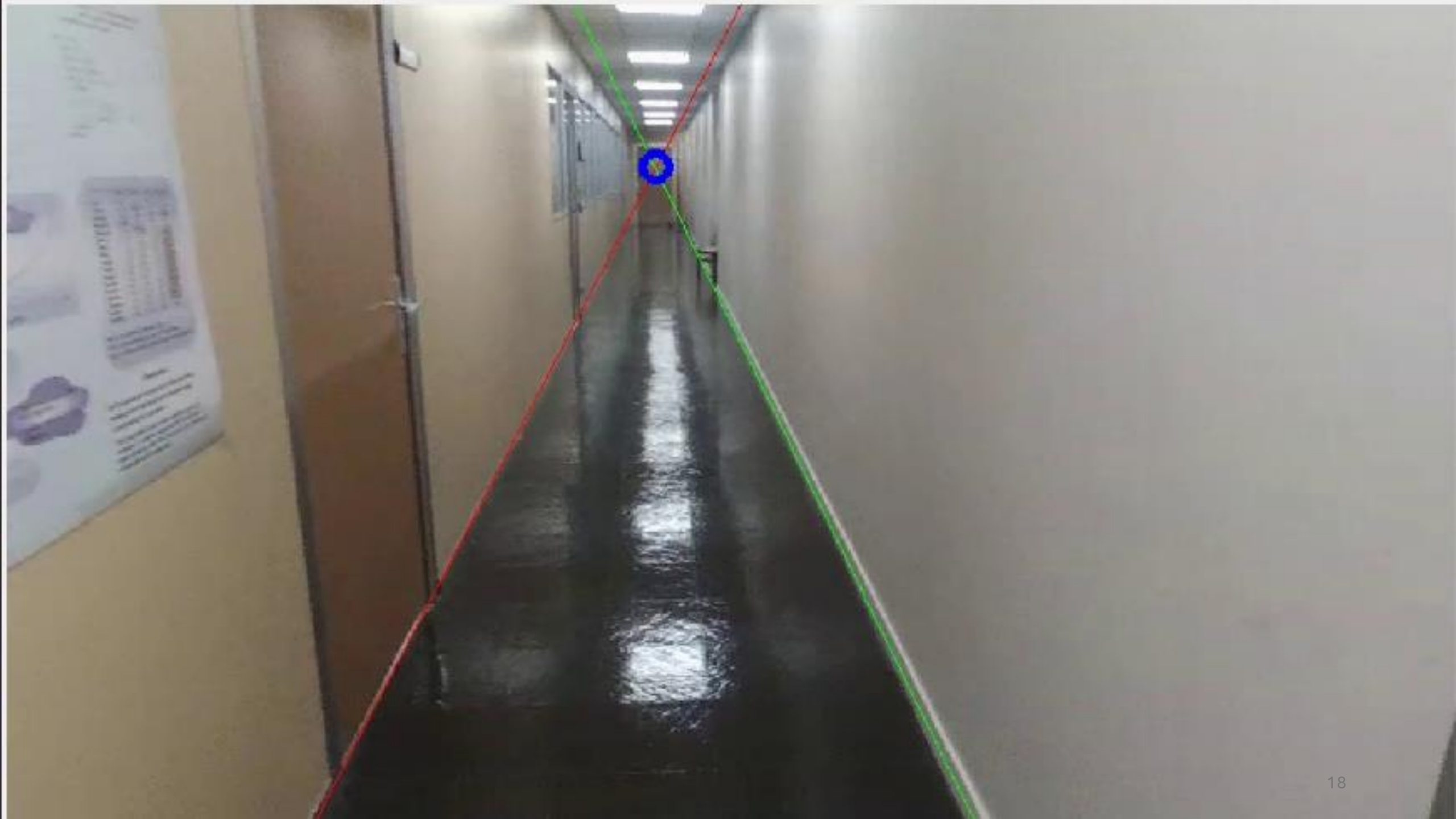
Un coup de Clustering

---

Mixer le tout

---

On a notre point de fuite

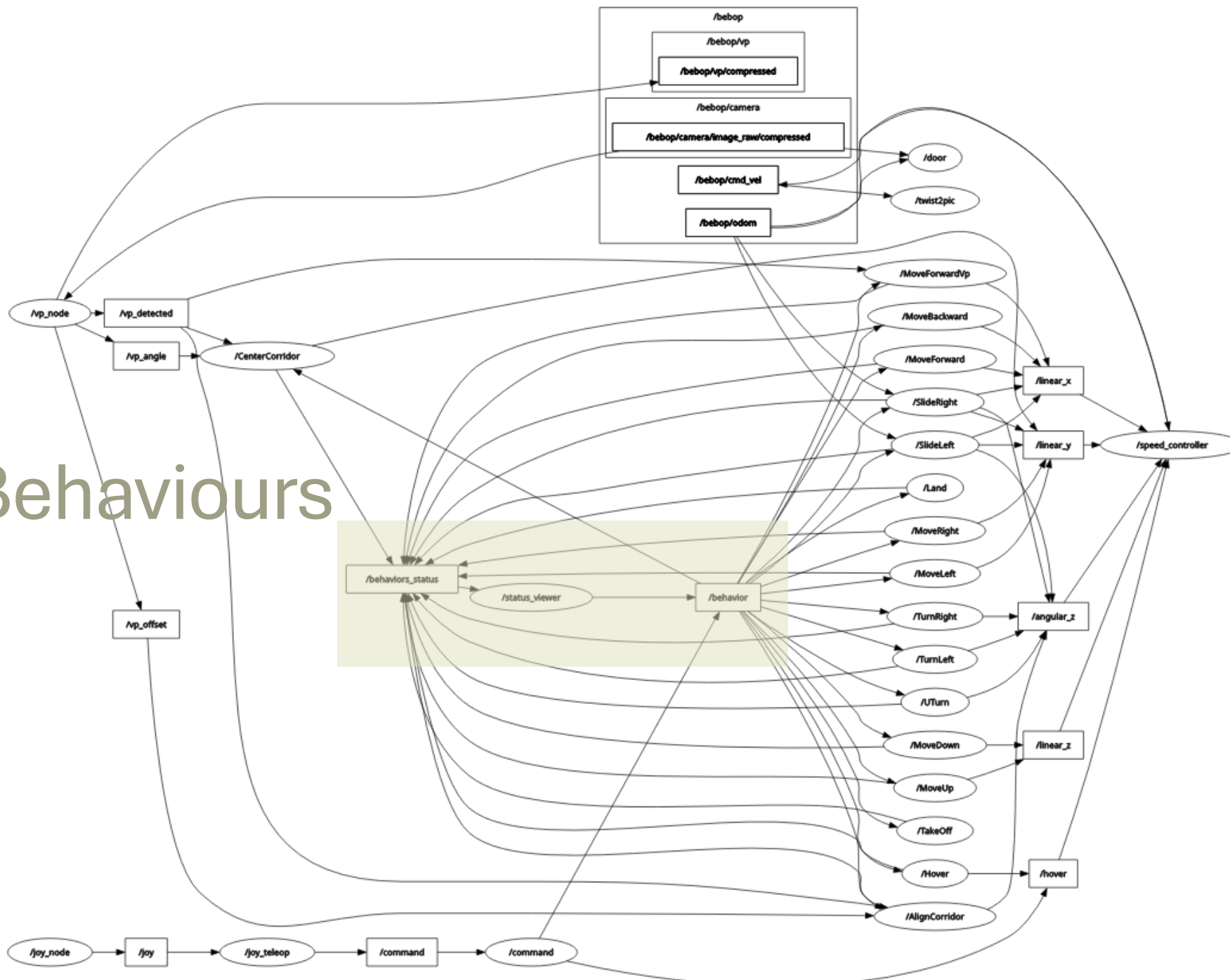


## Bebop, le Séducteur des Fenêtres

- On observe durant nos test qu'en présence de fenêtres, le bebop est attiré par celles-ci.



# Behaviours





toto - rqt

File Plugins Running Perspectives Help

Image View

/bebop/camera/image\_raw/compressed



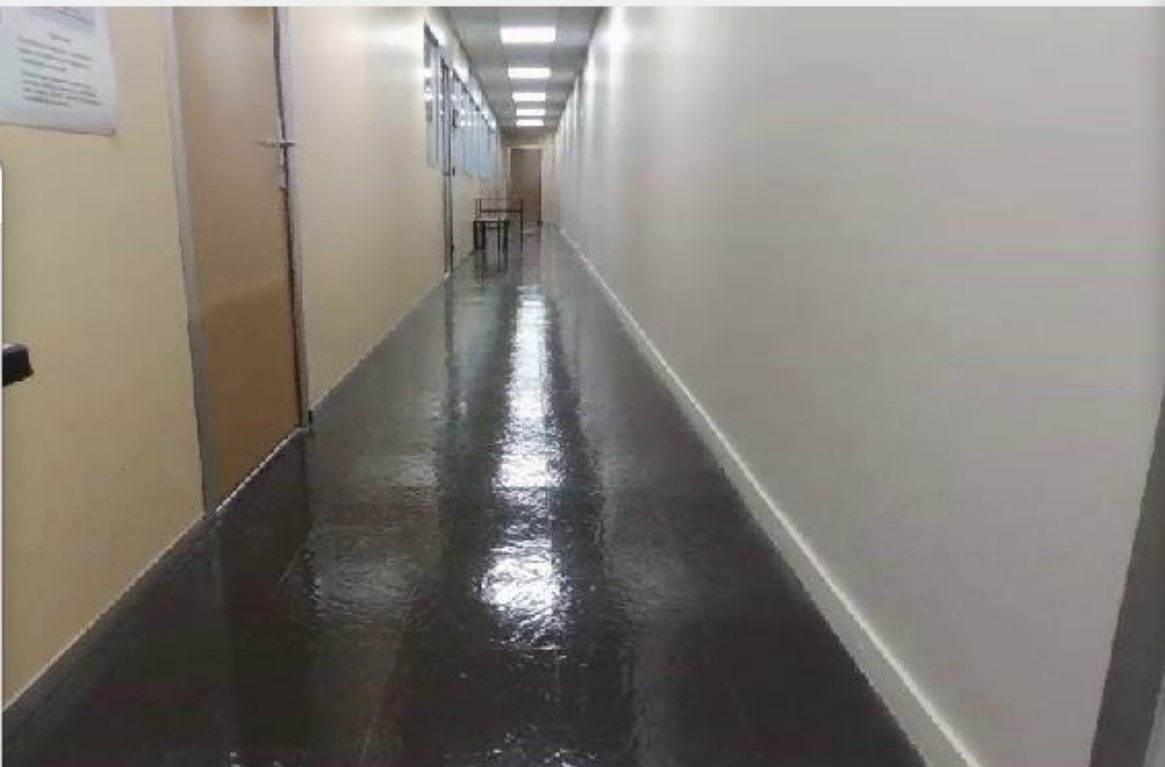
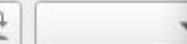
0



10.00m

☐ /camera/image\_raw/compressed\_mouse\_left☐ Smooth scaling

0°



Behaviors

- TakeOff
- AlignCorridor
- MoveUp
- Land
- MoveRight
- MoveBackward
- Hover
- MoveLeft
- CenterCorridor
- TurnLeft
- MoveForward
- TurnRight
- MoveDown

&gt; visual\_processing

① README.md

&gt; OUTLINE

rqt\_image\_view\_ImageView - rqt

Image View

/image\_out/compressed



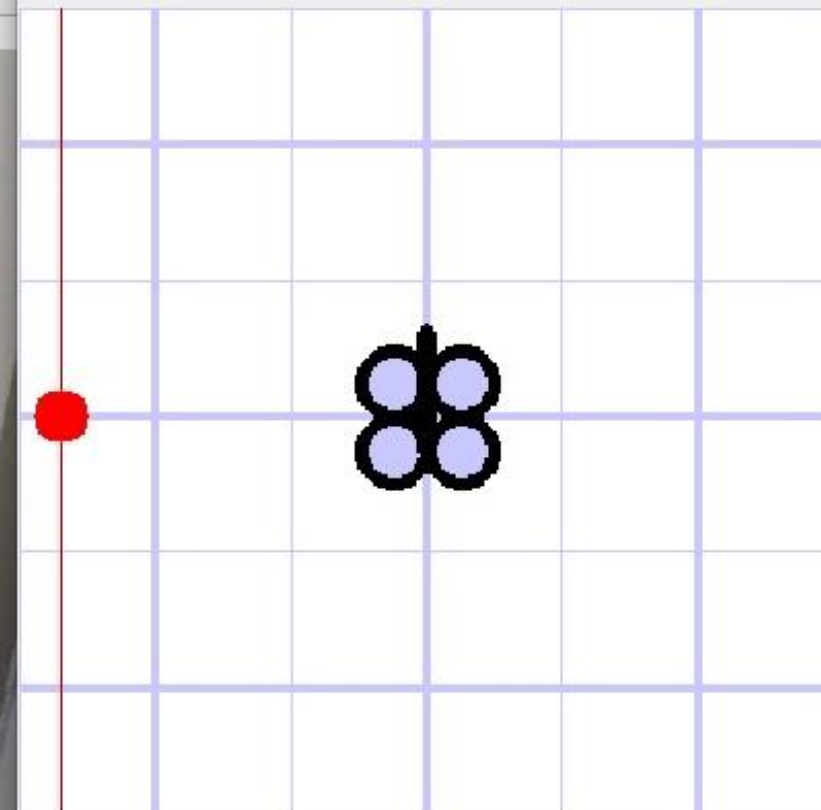
0



10.00m

☐ ed\_mouse\_left☐ Smooth scaling

0°



dans la frame

```
[vp_node-4] [INFO] [1730965840.506227300] [vp_node]: Receiving video frame (480, 856, 3), of type : <class 'numpy.ndarray'>
```

```
st5drone@st5drone-Latitude-5490: ~/Desktop/EI_drone$ git pull
```

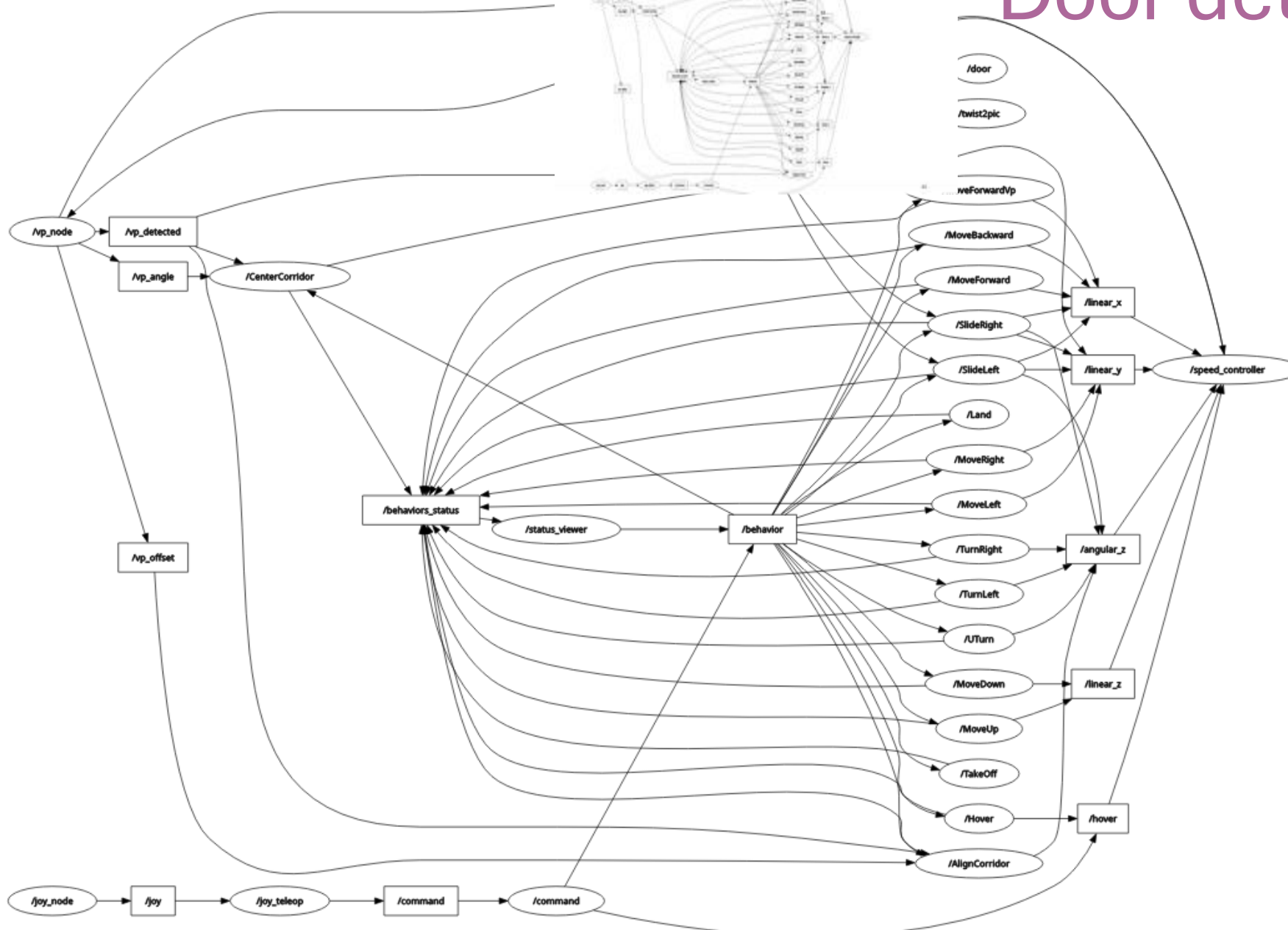
Already up to date.

```
• st5drone@st5drone-Latitude-5490:~/Desktop/EI_drone$ cd ..
```

```
• st5drone@st5drone-Latitude-5490:~/Desktop$ cd Mars_workspace/
```

Mars\_workspace + ▾ □ ✕ ... ^

# Door detection



# Optical flow detection

$$\text{shift}(x) = \underset{s \in [0, \sigma]}{\operatorname{argmin}} \sum_{h=-\rho}^{h=\rho} (i(x+h) - i'(x+h-s))^2$$

Fenêtre coulissante pour la variance des signaux

$$d = \gamma \frac{v}{f}$$

On filtre:

- Les shifts trop hauts
- Les signaux à faible contraste
- Les portes pas assez larges





