Supplementary Material: Navigation Formulas

Chasing Millimeters: Design, Navigation and State Estimation for Precise In-flight Marking on Ceilings

1 Coordinate frames

For convenience, we reproduce the coordinate frame table from the main paper here.

Name	Abbr.	Parent	Fixed	Source
Building	$\mathcal{B}$	-	✓	-
Odometry	$\mathcal{O}$	$\mathcal{B}$		Global estimate
IMU	$\mathcal{I}$	$\mathcal{O}$		Local estimate
VIO sensor	$\mathcal{O}_S$	$\mathcal{I}$	✓	CAD model
Prism	$\mathcal{P}$	$\mathcal{I}$	✓	CAD model
Tool sensor	$\mathcal{T}_S$	$\mathcal{I}$	✓	Calibration
Tool	$\mathcal{T}$	$\mathcal{T}_S$		Tracking

Table 1 All coordinate frames used by the full system including their abbreviation, parent frame, whether they are fixed or not, and how the transformation to their parent frame was obtained

2 Notation

f	Acceleration function of a policy
A	Metric of a policy
$(f, A)_{\mathcal{X}}$	Policy formulated in manifold $\mathcal{X}$
$\mathcal{P}_{\mathcal{X}}^{eef}$	Shorthand for policy (f, A) named eef formulated in $\mathcal{X}$
$\mathbf{T}_{ab}$	Full coordinate transform from frame b to a
$\mathbf{R}_{ab}$	Rotational part of $\mathbf{T}_{a,}b$
${}_a\boldsymbol{p}_x$	Point x expressed in frame a
${}_a\boldsymbol{v}_b$	Velocity of origin of frame b expressed in a.
${}_{\mathcal{B}}\boldsymbol{p}_{\mathcal{T}}$	Position of tool frame origin in Building frame
${}_{\mathcal{B}}\boldsymbol{p}_{\mathcal{T}}$	Position of tool frame origin in Building frame
${}_{\mathcal{B}}\boldsymbol{p}_{\otimes}$	Target marking position in Building frame

3 Helper functions

- Soft-normalization function (1)

$$\mathbb{S}(z) = \frac{z}{\mathsf{z} + \gamma \; log(1 + exp(\gamma \mathsf{z}))} \tag{1}$$

with tuning parameter γ.

4 Policies

4.1 End-effector navigation

- Index: een
- Defined on $\mathcal{B}$
- Default tuning:
  - $\alpha_{een} = 30$
  - $\beta_{een} = 10$
  - $\gamma_{een} = 0.1$

$$f_{een} = \alpha_{een} \cdot \mathbb{S}({}_{\mathcal{B}}\boldsymbol{p}_{\mathcal{T}} - {}_{\mathcal{B}}\,\boldsymbol{p}_{\otimes}) - \beta_{een}\,{}_{\mathcal{B}}\boldsymbol{v}_{\mathcal{T}} \tag{2}$$

$$A_{een} = diag([1, 1, 0, 0, 0, 0]) \tag{3}$$

$${}_{\mathcal{B}}\boldsymbol{p}_{\mathcal{T}} = \mathbf{T}_{\mathcal{B}\mathcal{O}}\mathbf{T}_{\mathcal{O}\mathcal{I}}\mathbf{T}_{\mathcal{I}\mathcal{T}_{\mathcal{S}}}\mathbf{T}_{\mathcal{T}_{\mathcal{S}}\mathcal{T}}\ {}_{\mathcal{T}}\boldsymbol{p}_{\mathcal{T}} \tag{4}$$

4.2 Depth servoing

- Index: ds
- Defined on $\mathcal{I}$
- Default tuning:
  - $\alpha_{ds} = 1$
  - $\beta_{ds} = 8$
  - $\gamma_{ds} = 0.05$

$$f_{ds} = \alpha_{ds} \cdot \mathbb{S}(D_{meas} - D_{des}) - \beta_{ds}\,{}_{\mathcal{I}}\boldsymbol{v}_{\mathcal{O}} \tag{5}$$

$$A_{ds} = diag([0, 0, 1, 0, 0, 0]) \tag{6}$$

- $D_{meas} \in R^6 = [0, 0, d_{meas}, 0, 0, 0]$, where d_{meas} is the measured distance to the ceiling as output by a depth sensor
- $D_{des} \in R^6 = [0, 0, d_{des}, 0, 0, 0]$, where d_{des} is the desired distance to ceiling in camera frame

4.3 Spring loading

- Index: sl
- Defined on $\mathcal{I}$
- Default tuning:
  - $\alpha_{sl} = 0.1$
  - $\gamma_{sl} = 0.1$

$$f_{ds} = \alpha_{sl} \cdot \mathbb{S}(S_{meas} - S_{des}) \tag{7}$$

$$A_{ds} = diag([0, 0, \tau_{ds}, 0, 0, 0]) \tag{8}$$

$$\tau_{ds} = 1.0/(1.0 + exp((\lambda - \lambda_{max}) * 25)) \tag{9}$$

- $S_{meas} \in R^6 = [0, 0, s_{meas}, 0, 0, 0]$, where s_{meas} is the measured spring extension, equivalent to the z value of ${}_{\mathcal{T}_S}\boldsymbol{p}_{\mathcal{T}}$
- $S_{des} \in R^6 = [0, 0, s_{des}, 0, 0, 0]$, where d_{des} is the desired spring extension.
- λ is the current amount of set-point deviation that causes impedance, i.e. how deep the flying base setpoint is inside the ceiling such that the impedance controller exerts a force.
- λ_{max} is the maximum permissible amount of set-point deviation

4.4 Prism tracking

- Index: pt
- Defined on $\mathcal{B}$
- Default tuning:
  - $\alpha_{pt} = 7$
  - $\beta_{pt} = 25$
  - $\gamma_{pt} = 0.05$

$$f_{pt} = \alpha_{pt} \cdot \mathbb{S}(\Psi prism) - \beta_{pt}\, {}_{\mathcal{I}}\boldsymbol{v}_{\mathcal{B}} \tag{10}$$
$$A_{pt} = diag([0, 0, 0, 0, 0, 1]) \tag{11}$$

- $\Psi_{prism} \in R^6 = [0, 0, 0, 0, 0, \psi_{prism}]$, where ψ_{prism} is the distance of the prism to the imaginary gravity-aligned plane between total station ${}_{\mathcal{B}}\boldsymbol{p}_{total}$ and the flying base center ${}_{\mathcal{B}}\boldsymbol{p}_{\mathcal{I}}$, calculated as:

$$\boldsymbol{a} = [0, 0, 1.0] \tag{12}$$
$$\hat{\boldsymbol{b}} = {}_{\mathcal{B}}\,\boldsymbol{p}_{\mathcal{I}} - {}_{\mathcal{B}}\,\boldsymbol{p}_{total} \tag{13}$$
$$\boldsymbol{b} = \hat{\boldsymbol{b}} / \|\hat{\boldsymbol{b}}\| \tag{14}$$
$$\boldsymbol{n} = \boldsymbol{a} \times \boldsymbol{b} \tag{15}$$
$$\boldsymbol{v} = \mathbf{T}_{\mathcal{BI}}\, {}_{\mathcal{I}}\boldsymbol{p}_{prism} - {}_{\mathcal{B}}\,\boldsymbol{p}_{\mathcal{I}} \tag{16}$$
$$\psi_{prism} = \boldsymbol{v} \cdot \boldsymbol{n} \tag{17}$$
$$\tag{18}$$

4.5 End-Effector following

- Index: eef
- Defined on $\mathcal{I}$
- Default tuning:
  - $\alpha_{eef} = 6$
  - $\beta_{eef} = 8$
  - $\gamma_{eef} = 0.005$

$$f_{eef} = \alpha_{eef} \cdot \mathbb{S}(\Delta_{eef} - \mathbf{0}) - \beta_{eef}\, {}_{\mathcal{O}}\boldsymbol{v}_{\mathcal{I}} \tag{19}$$

$$A_{eef} = diag([1, 1, 0, 0, 0, 0]) \tag{20}$$
$$\tag{21}$$

- $\Delta_{eef} \in R^6 = [\mathbf{R}_{\mathcal{I}\mathcal{T}_\mathcal{S}}\, {}_{\mathcal{T}_\mathcal{S}}\boldsymbol{p}_\mathcal{T}, 0, 0, 0]$, where $\mathbf{R}_{\mathcal{I}\mathcal{T}_\mathcal{S}}\, {}_{\mathcal{T}_\mathcal{S}}\boldsymbol{p}_\mathcal{T} \in R^3$ is the position of the end-effector w.r.t to the flying base.

5 Summation and Execution

5.1 Policy summation

Policies are summed using the sum operator from (1):

$$\begin{aligned}(\bar{f}, \bar{A}) &= sum(\mathcal{P}^0, \ldots, \mathcal{P}^n) \\ &= \left(\left(\sum_{i=0}^{n} A^i \right)^+ \sum_{i=0}^{n} A^i f^i, \sum_{i=0}^{n} A^i \right).\end{aligned}$$

5.2 Policy transformation

Policies are transformed (pulled) from one space as defined in (1):

$$pull_{\mathcal{C}\leftarrow\mathcal{X}}((f, A)_\mathcal{X}) = ((J^T A J)^+ J^T A f, J^T A J)_\mathcal{C}, \tag{22}$$

with J being the Jacobian relating $\mathcal{X}$ and $\mathcal{C}$ locally. As all coordinate frames used here are equivalent to $SE(3)$, J is simply the rotation from $\mathcal{X}$ to $\mathcal{C}$, $\mathbf{R}_{\mathcal{C}\mathcal{X}}$.

5.3 End-Effector

The velocity command C_{ee} to the end-effector is calculated as:

$$C_{ee} = \int_0^{\delta t} f_{een} \quad dt \tag{23}$$

of which only the lateral x, y and the rotational yaw velocity are used. Note that f_{een} implicitly is a function of the time t. δt is 0.005 s.

5.4 Flying base

The velocity command C_{fb} to the flying base is calculated as:

$$\mathcal{P}_\mathcal{I}^{body} = sum(\{\mathcal{P}^{ds}, \mathcal{P}^{sl}, \mathcal{P}^{eef}\}) \tag{24}$$
$$\mathcal{P}_\mathcal{O}^{body} = pull_{\mathcal{O}\leftarrow\mathcal{I}}(\mathcal{P}_\mathcal{I}^{body}) \tag{25}$$
$$\mathcal{P}_\mathcal{O}^{global} = pull_{\mathcal{O}\leftarrow\mathcal{B}}(\{\mathcal{P}^{pt}\}) \tag{26}$$
$$\bar{\mathcal{P}} = (\bar{f}, \bar{A}) = sum(\{\mathcal{P}_\mathcal{O}^{body}, \mathcal{P}_\mathcal{O}^{global}\}) \tag{27}$$

$$C_{fb} = \int_0^{\delta t} \bar{f} \quad dt \tag{28}$$

Note that $\bar{f}$ implicitly is a function of the time t. δt is 0.005 s.

References

[1] Ratliff, Nathan D and Issac, Jan and Kappler, Daniel and Birchfield, Stan and Fox, Dieter (2018) *Riemannian motion policies*, *arXiv preprint*, DOI: 10.48550/ARXIV.1801.02854