A bootstrappable, bio-plausible design for visual pose stabilization

Andrea Censi, Shuo Han, Sawyer Fuller, Andrew Straw, Richard Murray
California Institute of Technology

This talk includes unpublished results -- see our recent draft at:
www.cds.caltech.edu/~hanshuo/
www.cds.caltech.edu/~andrea/
What engineering can learn from neurobiology

- Fast behavior from slow computational elements.
- Robust behavior from noisy processing.
- Coherent decisions from extremely distributed computation.
- Data fusion from disparate senses.
- Learning
- Consciousness
Visual processing in the fruit fly

- A well studied example of biological control architecture.
- We know something about visual information processing, but we miss both the details (role of each neuron) and the big picture (integrative model).

What is remarkable
- **Slow** computation, **fast** reaction
  - time constant of photoreceptors is comparable to reaction time.
- **Noisy** sensors/computation, **robust** reaction
- **distributed** computation using neurons.

Can we design control laws for such an architecture?
Visual pose stabilization (servoing, homing)

- Drive the agent from the current image $y$ to a given goal image $g$. 

![Diagram showing the process of visual pose stabilization](image)

- Controller
  - Input: goal image $g$
  - Output: force, torque $f, \tau$

- Rigid body dynamics
  - Input: $f, \tau$
  - Output: pose, velocities $r, p, \omega, \nu$

- Vision sensor
  - Input: pose, velocities $r, p, \omega, \nu$
  - Output: current image $y$, $\dot{y}$

- World geometry
  - Input: current image $y$
  - Output: goal image $g$
Variations of the same problem

visual attitude stabilization
(pure rotational case)

hovering
(goal image is delayed image)

visual homing
(memorized goal image is “far”)

visual navigation
Fontanelli et al., 2009
### Notations

**Robot:** fully actuated, 2nd-order system
- Pose (attitude + position): $q = (r, p) \in \text{SE}(3) = \text{SO}(3) \times \mathbb{R}^3$ (strictly, $\text{SO}(3) \otimes \mathbb{R}^3$)
- Body velocities: $(\omega, v)$
- Control inputs: torque and force $(\tau, f)$

**Sensor:** an omnidirectional camera
- Direction: on a unit sphere $s \in \mathbb{S}^2$
- Goal image (intensity profile): $g(s) : \mathbb{S}^2 \to \mathbb{R}$
- Current image: $y(s) : \mathbb{S}^2 \to \mathbb{R}$

**Environment:** convex
- Distance to the objects: $\sigma(s) : \mathbb{S}^2 \to \mathbb{R}$
- Its reciprocal, "nearness", is sometimes used:
  \[ \mu(s) \triangleq 1/\sigma(s) \]
- Nearness is an hidden state in the dynamics of $y$:
  \[ \dot{y} = \mu(s) \nabla y^* v + (Sy)^* \omega \]
Solutions #1: State estimation
- Estimate the current pose $q$ (by finding the best match between $y$ and $g$)
  $$q = \arg \min q \| y - g \circ q \|^2$$
  Rotated + translated version of $g$
- Then drive the agent towards the goal:
  $$q \xrightarrow{\xi} q_g$$
- Not always possible: needs to know the geometry of the environment, including the nearness.

Solution #2: point features in image space
- Extract point features $s_i$
- Solve correspondences $s_i$ (current image) $\leftrightarrow s_i^*$ (goal image)
- Control law: depends on $(s_i - s_i^*)$

[Booij et al, ICRA'07]

• Combinatorial problem not implementable on a neural substrate.
Feature-free image space approach

Control law based on images directly

Minimize the following metric in the image space:

\[ J(q) = \frac{1}{2} \langle (y - g)^2 \rangle \]

"\[ \langle \ldots \rangle \]" denotes integral on the sphere

- The gradient flow for this cost function depends on the nearness \( \mu(s) \)

Related work

- Kernel-based visual servoing (Kallem, IROS’07)
- Different descent methods (Collewet, ICRA’08)
- Using mutual information (Dame, ICRA’09)

What they did not show

- Rigorous convergence proofs
  - Is minimizing \( J \) the same as going to the goal pose?
  - Can we do it without the knowledge of \( \mu(s) \) (environment geometry)?
- Control of a second-order system (inputs are torques and forces)
Question: is $y = g$ equivalent to $\text{pose} = \text{goal pose}$?

- Sufficient condition: the metric $J$ is strictly convex near the goal

$$J(q) = \frac{1}{2} \langle (y - g)^2 \rangle$$

- Can be determined by requiring the $6 \times 6$ Hessian matrix to be positive definite

$$C(y, \mu) = \begin{bmatrix}
\langle SySy^* \rangle & \langle \mu Sy \nabla y^* \rangle \\
\langle \mu \nabla y Sy^* \rangle & \langle \mu^2 \nabla y \nabla y^* \rangle
\end{bmatrix} > 0 \quad Sy \triangleq s \times \nabla_s y$$

- This means the observation will change in every direction of motion.
Theorem: For the 2nd-order rigid body system:

\[
\begin{align*}
\dot{r} &= r (\omega)^\wedge, \\
\Pi \dot{\omega} &= (\Pi \omega) \times \omega - \epsilon \omega \omega + \tau, \quad \epsilon_{\omega}, \epsilon_{v} \geq 0 \\
\dot{p} &= rv, \\
m \dot{v} &= m v \times \omega - \epsilon_{v} v + f
\end{align*}
\]

the following control law is locally asymptotically stable \((y \to g, q \to q_{g})\)

\[
\begin{align*}
\tau &= \langle (Sy) (g - y) \rangle - k_{d} \langle (Sy) \dot{y} \rangle, \\
f &= \alpha \langle (\nabla y) (g - y) \rangle - \alpha k_{d} \langle (\nabla y) \dot{y} \rangle
\end{align*}
\]

if the modified contrast condition holds

\[
\begin{bmatrix}
\langle Sy Sy^* \rangle \\
\langle (\frac{\alpha}{2} + \frac{\mu}{2}) \nabla y Sy^* \rangle \\
\langle (\frac{\alpha}{2} + \frac{\mu}{2}) \nabla y Sy^* \rangle \\
\alpha \langle \mu \nabla y \nabla y^* \rangle
\end{bmatrix} > 0
\]

Note: the CDC paper proves the case of pure rotation. For SE(3), see our submission to ICRA’10, available on our websites.
The PD control law

Proportional-derivative control law:

\[
\begin{align*}
\tau &= \langle (Sy)(g - y) \rangle - k_d \langle (Sy) \dot{y} \rangle, \\
f &= \alpha \langle (\nabla y)(g - y) \rangle - \alpha k_d \langle (\nabla y) \dot{y} \rangle
\end{align*}
\]

- \( S \) and \( \nabla \) are differential operators (on the sphere)

**Proportional part:**
- similar to the gradient of the quadratic metric
- \( J(q) = \frac{1}{2} \langle (y - g)^2 \rangle \)
- but without nearness.

**Derivative part:**
- similar to the least-squares velocity estimation from \( y \) and \( \dot{y} \)
- \( \dot{y} = \mu(s) \nabla^* v + (Sy)^* \omega \)
- but without the matrix inverse.

**Comments**
- No explicit (point) feature extraction
- Does not depend on nearness.
- Bilinear/quadratic in \( g, y, \) and \( dy/dt \).
- Sparse tensors \( A, B \)

**Diagram**

Tensor contraction:

\[
\begin{align*}
\tau^k &= A^u_k w^v y_u, \\
f^k &= B^u_k w^v y_u
\end{align*}
\]
Visual simulation: a software package ‘fsee’ (by A. Straw)

- Based on the fruit fly Drosophila
- Correct disposition of the 1400 “pixels” (ommatidia) of the compound eye
- Spatial Gaussian blurring
- Temporal response of photoreceptors
Convergence results

Example run

Convergence statistics

Still get > 50% convergence for initial errors = (30 deg, 3 m)
Bootstrapping the control law

The learning problem

- The control strategy is a tensor product for some $n \times n \times 3$ tensors $A, B$.
- Can we learn $A, B$?

Why learning?

- **Engineering**: writing $A, B$ explicitly requires to know the intrinsic calibration of the camera; learning directly $A, B$ bypasses camera calibration.
- **Biology**: encoding a learning law is cheaper than encoding $n \times n \times 3$ coefficients.

How? We found a very simple solution based on unsupervised Hebbian learning:

- Generate random forces and torques.
- Learn $A, B$ using Hebbian learning of $y, \dot{y}$, and $(\tau, f)$

Note: the CDC paper uses supervised backpropagation.
Conclusions

We solved the problem of visual stabilization on SE(3) on a bio-plausible architecture

- without extracting features
- we proved it works even though distance is not observable
- parallelizable, simple control law respects the constraints of a bio-plausible computation
  - and also GPU computation!
- the control law can be bootstrapped (learned unsupervisedly)
  - allows auto-calibrating system.

Current research directions

- Same approach for more complicated problems, such as obstacle avoidance.
- Using these models for understanding the behavior of fruit flies.

Note: visit our websites for draft containing the results for SE(3):
www.cds.caltech.edu/~hanshuo/
www.cds.caltech.edu/~andrea/