Title

Montiel Abello

Chapter 1

Introduction

1.1 Literature Review

1.1.1 infinite-dimensional observers

linear:
-observer theory for linear infinite-dimensional systems widely studied
-techniques used typically extensions of luenberger observers used for finite-dimensional systems.
-simplified approach: use spatial discretisation such as finite different/finite element to reduce infinite-dimensional to finite-dimensional observer. [1, 2]
-better to design infinite-dimensional observer and only discretise for numerical implementation. [3, 4, 5] **TODO:** *describe these design methods.*
nonlinear:
-no universal approach for observer design for infinite-dimensional nonlinear systems
-some methods for special case - infinite dimensional bilinear systems. [6, 7]
-for finite-dimensional nonlinear systems, common design methods are: linearisation (ie EKF), lyapunov method, sliding mode, high gain

1.1.2 symmetry preserving observers

1.1.2.1 early work

1.1.2.2 bonnabel et al

1.1.2.3 trumpf, mahony et al

1.1.2.4 juan's work - in detail

1.2 Theoretical Background

1.2.1 Rigid Body Dynamics

1.2.1.1 Lie Groups

A Lie group $\mathbf{G}$ is a group that is also a differentiable manifold. As a group, $\mathbf{G}$ is a set of elements and a group operation (denoted by multiplication, i.e. AB for $A, B \in \mathbf{G}$) that satisfies the 4 group axioms:

- **Closure:** The group operation $\mathbf{G} \times \mathbf{G} \mapsto \mathbf{G}$ is a function that maps elements of $\mathbf{G}$ onto itself; $\forall A, B \in \mathbf{G}$, $AB \in \mathbf{G}$.
- **Associativity:** Elements of G are associative under the group operation; $\forall A, B, C \in \mathbf{G}$, $(AB)C = A(BC)$.
- **Identity:** There exists an identity element $I \in \mathbf{G}$ such that $\forall A \in \mathbf{G}$, $IA = AI = A$.
- **Inverse:** For all $A \in \mathbf{G}$ there exists an inverse element $A^{-1} \in \mathbf{G}$ such that $AA^{-1} = A^{-1}A = I$.

Because the Lie group $\mathbf{G}$ is a differentiable manifold, it is locally Euclidean. This means that the neighbourhood around every element of $\mathbf{G}$ can be approximated with a tangent plane. This property allows calculus to be performed on elements of $\mathbf{G}$.

Matrix Lie groups
A matrix Lie group is made up of group elements which are $n \times n$ matrices. This work will be focus on matrix Lie groups because the exponential map and Lie bracket functions given below only apply to such Lie groups.

Lie algebra
The tangent space at the identity element of a Lie group is called the Lie algebra $\mathfrak{g}$. It is called the Lie *algebra* because it has a binary operation, known as the Lie bracket $[X, Y]$. For matrix Lie groups the Lie bracket is

$$[A, B] \triangleq AB - BA \tag{1.1}$$

The exponential map and logarithm map
The mapping from the Lie algebra $\mathfrak{g}$ to the Lie group $\mathbf{G}$ is called the exponential map:

$$\exp : \mathfrak{g} \to \mathbf{G} \tag{1.2}$$

Similarly, the logarithm map maps elements from $\mathbf{G}$ to $\mathfrak{g}$:

$$\log : \mathbf{G} \to \mathfrak{g} \tag{1.3}$$

sub-heading?
For an n-dimensional matrix Lie group, the Lie algebra $\mathfrak{g}$ is a vector space isomorphic to $\mathbb{R}^n$. The hat operator $\hat{}$ maps vectors $x \in \mathbb{R}^3$ to elements of $\mathfrak{g}$.

$$\hat{} : x \in \mathbb{R}^n \to \hat{x} \in \mathfrak{g} \tag{1.4}$$

For a matrix Lie group $\mathbf{G}$ whose elements are $n \times n$ matrices, the elements of $\mathfrak{g}$ will also be $n \times n$ matrices. The hat operator is defined

$$\hat{x} = \sum_{i=1}^{n} x_i G^i \tag{1.5}$$

where G^i are $n \times n$ matrices known as the infinitesimal generators of $\mathbf{G}$.

Lie bracket and group operation
For Lie groups endowned with the commutative property ($\forall A, B \mathbf{G}, AB = BA$), vector addition in the Lie algebra maps to a group operation in the Lie group. For $C = A + B$ where $A, B, C \in \mathfrak{g}$,

$$e^C = e^{A+B} = e^A e^B \tag{1.6}$$

For non-commutative Lie groups, the relationship between the Lie bracket and group operation does not hold. Instead, for $C = \log e^A e^B$, C is calculated with the Baker-Campbell-Hausdorff formula:

$$C = A + B + \frac{1}{2}[A, B] + \frac{1}{12}[A - B, [A, B]] + \frac{1}{24}[B, [A, [A, B]]] + \dots \tag{1.7}$$

Actions
When a group action for a Lie group G acting on a manifold M is a differentiable map, this is known as a Lie group action. For example, 3D rotations act on 3D points so the Lie group $\mathbf{SO}(3)$ acts on $\mathbb{R}^3$. A left action of $\mathbf{G}$ on M is defined as a differentiable map

$$\Phi : \mathbf{G} \times M \mapsto M \tag{1.8}$$

where

- the identity element I maps M onto itself *(is that the right wording?)

$$\Phi(I, m) = m, \forall m \in M \tag{1.9}$$

- Group actions compose according to

$$\Phi(m, \Phi(n, o)) = \Phi(mn, o) \tag{1.10}$$

Adjoint map
EXPLANATION - Sometimes before a function B can be applied on manifold acted on by a group action A, it is necessary to apply the conjugate of A to B????
For $A \in \mathbf{G}$ define a function Ψ, known as the adjoint map of $\mathbf{G}$:

$$\Psi_A : \mathbf{G} \to \mathbf{G},\ \Psi_A(B) \triangleq ABA^{-1} \tag{1.11}$$

Taking the derivative:

$$\frac{\partial}{\partial t}\Psi_A(B(t))|_{t=0} = AVA^{-1},\ V \triangleq \frac{\partial}{\partial t}B(t)|_{t=0} \tag{1.12}$$

The adjoint representation of $\mathbf{G}$ is given by the mapping

$$\mathbf{Adj}_A : \mathfrak{g} \to \mathfrak{g},\ \mathbf{Adj}_A(V) \triangleq AVA^{-1} \tag{1.13}$$

1.2.1.2 SO(3)

A rotation represents the motion of a point about the origin of a Euclidean space. In $\mathbb{R}^3$ this is a proper isometry: a transformation that preserves distances between any pair of points and has a determinant of $+1$. The set of all rotations about the origin of $\mathbb{R}^3$ is known as the *special orthogonal group* $\mathbf{SO}(3)$. This matrix Lie group is a subgroup of the general linear group $\mathbf{GL}(3)$, so its group elements are 3×3 invertible matrices. These group elements are orthogonal matrices so their columns and rows are orthogonal unit vectors.

Lie algebra
Lie algebra $\mathfrak{so}(3)$ is vector space of 3×3 skew-symmetric matrices $\hat{\omega}$, where ω is a 3-vector representing an angular velocity. The direction of ω indicates the axis of rotation while its magnitude gives the angular velocity. Elements of $\mathfrak{so}(3)$ are mapped to $\mathbf{SO}(3)$ according to the exponential map:

$$\begin{aligned} \exp : \mathfrak{so}(3) &\to \mathbf{SO}(3) \\ [\omega]_\times &\to \mathbf{R}_{3x3} \end{aligned} \tag{1.14}$$

i.e. $\forall \omega \in \mathfrak{so}(3),\ \exp([\omega]_\times) \in \mathbf{SO}(3)$

Conversely, the logarithm map maps 3×3 rotation matrices of $\mathbf{SO}(3)$ to elements of $\mathfrak{so}(3)$:

$$\begin{aligned} \log : \mathbf{SO}(3) &\to \mathfrak{so}(3) \\ \mathbf{R}_{3x3} &\to [\omega]_\times \end{aligned} \tag{1.15}$$

i.e. $\forall \mathbf{R} \in \mathbf{SO}(3),\ \log(\mathbf{R}) \in \mathfrak{so}(3)$

Actions
By the group action, elements of $\mathbf{SO}(3)$ rotate points in $\mathbb{R}^3$ about the origin.

$$\Phi : \mathbf{SO}(3) \times \mathbf{R}^3 \mapsto \mathbf{R}^3 \tag{1.16}$$

Adjoint map

$$\Psi_R : \mathbf{SO}(3) \to \mathbf{SO}(3),\ \Psi_R(A) \triangleq RAR^{-1} \tag{1.17}$$

Taking the derivative:

$$\frac{\partial}{\partial t}\Psi_R(A(t))|_{t=0} = RBR^{-1},\ B \triangleq \frac{\partial}{\partial t}A(t)|_{t=0} \tag{1.18}$$

The adjoint representation of $\mathbf{G}$ is given by the mapping

$$\mathbf{Adj}_R : \mathfrak{so}(3) \to \mathfrak{so}(3),\ \mathbf{Adj}_R(B) \triangleq RBR^{-1} \tag{1.19}$$

???? Hard to explain practical application without discussing reference frames: ie if position defined in body fixed frame but some other transformation defined in inertial frame. First undo rotation to get pose in inertial frame, apply transformation, then re-apply rotation

Rotation representation
There are many conventions by which elements of $\mathbf{SO}(3)$ can be represented. Those A rotation about a point in $\mathbb{R}^3$ can be represented by: **TODO:** *go into more detail on below*

Rotation matrix
3×3 matrix where magnitude of each column is 1, columns are orthogonal, determinant is $+1$. Group operation matrix is multiplication. Left action is left multiplication of point.

Scaled axis
3-vector where direction represents axis of rotation and magnitude represents angle of rotation. Group operation - vector addition? Left action - rodrigues' rotation formula

Quaternion
4-vector, same information as axis angle, but different form. Given axis of rotation $\mathbf{r}$ and angle of rotation θ:

$$\mathbf{q} = \begin{bmatrix} w \\ x \\ y \\ z \end{bmatrix} = \begin{bmatrix} w \\ \mathbf{v} \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) \\ \sin(\theta/2)\mathbf{r} \end{bmatrix} \tag{1.20}$$

Group operation - quaternion multiplication defined as:

$$\mathbf{q1q2} = \begin{bmatrix} w_1 \\ \mathbf{v}_1 \end{bmatrix} \begin{bmatrix} w_2 \\ \mathbf{v}_2 \end{bmatrix} = \begin{bmatrix} w_1 w_2 - \mathbf{v}_1 \cdot \mathbf{v}_2 \\ w_1 \mathbf{v}_2 + w_2 \mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2 \end{bmatrix} \tag{1.21}$$

As with rotation matrices, quaternion multiplication is associative but not commutative.

1.2.1.3 SE(3)

3D space - $\mathbb{R}^3$
In practice, robot, sensor, environment exist in 3D Euclidean space - $\mathbb{R}^3$.
An arbitrary function that maps a pose *(or point?) in $\mathbb{R}^3$ to another can be defined as:

$$f : \mathbb{R}^3 \to \mathbb{R}^3 \tag{1.22}$$

To represent rigid bodies, require mappings corresponding to rotation and translation. Translation can be modelled as a function on a vector space $\mathbb{R}^3$ but the set of all rotations in $\mathbb{R}^3$ forms a Lie group.

homogeneous representation
4×4 screw matrix - represent rotation and translation with a single matrix of form:

$$\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \tag{1.23}$$

TODO: align the R and 0!!!!

To apply a rigid transformation to a point $\mathbf{p} = (x, y, z)$ in $\mathbb{R}^3$, represent with homogeneous coordinates. ie

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1.24}$$

Elements of SE(3)

$$\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \tag{1.25}$$

Lie algebra

$$\begin{bmatrix} [\omega]_\times & \mathbf{v} \\ \mathbf{0}_{1\times 3} & 0 \end{bmatrix} \tag{1.26}$$

exponential map

Actions
group element acts on $\mathbb{R}^3$ - point in homogeneous coordinates performs rigid transformation

Adjoint Map
-adjoint map & adjoint representation -adjoint map changes frame that pose is defined in???

1.2.1.4 Reference Frames

A reference frame is a system used to define a point on a manifold, on this case the Euclidean space $\mathbb{R}^3$. A reference frame is represented by an element of $\mathbf{SE}(3)$.
${}^A_B\mathbf{X}_C$ defines transformation of C w.r.t. B defined in A

Definition: Pose

Definition: point -homogeneous coordinates

Inverse

$$({}^A_B\mathbf{X}_C)^{-1} = {}^A_C\mathbf{X}_B \tag{1.27}$$

Transform point from one reference frame to another:

$${}^A_A\mathbf{p}_B = {}^A_A\mathbf{X}_B {}^B_A\mathbf{p}_B \tag{1.28}$$

$${}^B_A\mathbf{p}_B = {}^B_B\mathbf{X}_A {}^A_A\mathbf{p}_B \tag{1.29}$$

Transform pose from one reference frame to another -change of basis

$${}^B_C\mathbf{X}_D = ({}^B_B\mathbf{X}_A){}^A_C\mathbf{X}_D({}^B_B\mathbf{X}_A)^{-1} \tag{1.30}$$

1.2.1.5 Sensor State Representation

-inertial frame A, sensor/robot frame B
-p,v,a,R,omega,alpha - define, state reference frames
position ${}^A_A\mathbf{p}_B$
velocity ${}^B_A\mathbf{v}_B$
acceleration ${}^B_A\mathbf{a}_B$
orientation ${}^A_A\mathbf{R}_B$ - rotation matrix
angular velocity ${}^B_A\omega_B$ - scaled axis representation
angular acceleration ${}^B_A\alpha_B$ - scaled axis representation

pose of robot w.r.t. inertial frame, defined in inertial frame ie. screw matrix:

$${}^A_A\mathbf{S}_B(t) = \begin{bmatrix} {}^A_A\mathbf{R}_B(t) & {}^A_A\mathbf{p}_B(t) \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \tag{1.31}$$

velocity (linear and angular) w.r.t. inertial frame, defined in body frame ie. twist matrix:

$$ {}^{B}_{A}\mathbf{T}_{B}(t) = \begin{bmatrix} [{}^{B}_{A}\omega_{B}(t)]_{\times} & {}^{B}_{A}\mathbf{v}_{B}(t) \\ \mathbf{0}_{1\times 3} & 0 \end{bmatrix} \tag{1.32} $$

acceleration (linear and angular) w.r.t. inertial frame, defined in body frame ie. wrench matrix

$$ {}^{B}_{A}\mathbf{W}_{B}(t) = \begin{bmatrix} [{}^{B}_{A}\alpha_{B}(t)]_{\times} & {}^{B}_{A}\mathbf{a}_{B}(t) \\ \mathbf{0}_{1\times 3} & 0 \end{bmatrix} \tag{1.33} $$

other parameters ie FOV, steps

1.2.1.6 Sensor Dynamic Model

screw matrix:

$$ \frac{\mathrm{d}}{\mathrm{d}t}{}^{A}_{A}\mathbf{S}_{B}(t) = {}^{A}_{A}\mathbf{S}_{B}(t){}^{B}_{A}\mathbf{T}_{B}(t) \tag{1.34} $$

twist matrix:

$$ \frac{\mathrm{d}}{\mathrm{d}t}{}^{B}_{A}\mathbf{T}_{B}(t) = {}^{B}_{A}\mathbf{W}_{B}(t) \tag{1.35} $$

wrench matrix:

$$ \frac{\mathrm{d}}{\mathrm{d}t}{}^{B}_{A}\mathbf{W}_{B}(t) = 0 \tag{1.36} $$

-something for scanning dynamics

-update methods (euler, runge-kutta etc)

1.2.1.7 Object State Representation

-frame fixed to object - B -same as sensor + size s

1.2.1.8 Object Dynamic Model

-ODEs (same as sensor)

1.2.2 Symmetry Preserving Observers

1.2.2.1 definitions?

1.2.2.2 construction, ie moving frame method etc

1.2.3 Infinite Dimensional Observers

1.2.4 Discretisation Methods?

1.3 Problem Statement

context: Advances in hardware and manufacturing have made autonomous and semi-autonomous robots more available. Use in industry and even general public has increased. Full autonomous robots are still limited to structured environments and tasks such as in factories and warehouses. Before robots can operate autonomously in unstructured environments, new sensor models are required to more effectively observe and represent complex environment states. **TODO:** *Why are new sensor models required? Check if this is covered in literature review. If not, need to expand on this.*

problem/lacking: One method of estimating the state of the environment is to use a state observer. The majority of observer implementations do not take into account the natural symmetries of the dynamics of the state. Doing so has shown to be beneficial in both the design of observers, and improved convergence properties. However, these invariant observer methods are still limited to finite dimensional systems. In many implementations involving infinite-dimensional systems, the system is discretised to a finite dimensional one prior to observer design. **TODO:** *How does this influence performance?*

What is needed is a theory of infinite dimensional, symmetry preserving observers, + design principles.

what will this theory provide?: This theory will simplify invariant observer design for infinite dimensional systems. Only discretising after observer design will maximise the potential of dense sensors. This will allow for more accurate and fast estimation of complex environments,

approach: This project aims to develop some of this theory. The approach taken will be to design an invariant observer for a specific infinite dimensional system, before generalising the results.

estimation problem - cube pose & size: The environment the robot is moving throughout consists of a room (rectangular prism) + cube; each of unknown size and pose. Attached to the robot is a 2D laser rangefinder. Using depth measurements from this sensor, the observer must estimate the size and pose of the cube. It is assumed that the room is stationary and the cube has constant angular and linear acceleration.
TODO: *Diagram*
TODO: *Precise mathematical description of problem*

deliverables:
The primary deliverable of this project is the observer design and simulation. Will later try to develop some general theory from this specific case. Will validate simulation with experiment using Hokuyo UBG 04-LX sensor, ???

robot arms and cubes of various sizes and materials. **TODO:** *More detail, + be careful about what you promise*

Chapter 2

Theory Results

Chapter 3

Simulation

3.1 Implementation

3.1.1 Sensor modelling

screw: ${}^{A}_{A}\mathbf{S}_{B}(t+\delta t) = {}^{A}_{A}\mathbf{S}_{B}(t)e^{\delta t {}^{B}_{A}\mathbf{T}_{B}(t)}$

twist: ${}^{B}_{A}\mathbf{T}_{B}(t+\delta t) = {}^{B}_{A}\mathbf{T}_{B}(t) + \delta t {}^{B}_{A}\mathbf{W}_{B}(t)$

wrench: ${}^{B}_{A}\mathbf{W}_{B}(t+\delta t) = {}^{B}_{A}\mathbf{W}_{B}(t)$

3.1.2 Environment modelling

3.1.3 Measurement modelling

-triangle ray intersection

3.1.4 Observer implementation

3.2 Results

Chapter 4

Experiment

Chapter 5

Conclusion

Bibliography

[1] C. Harkort and J. Deutscher, "Finite-dimensional observer-based control of linear distributed parameter systems using cascaded output observers," *International Journal of Control*, vol. 84, no. 1, pp. 107–122, 2011.

[2] L. Meirovitch and H. Baruh, "On the problem of observation spillover in self-adjoint distributed-parameter systems," *Journal of Optimization Theory and Applications*, vol. 39, no. 2, pp. 269–291, 1983.

[3] G. Haine, "Recovering the observable part of the initial data of an infinite-dimensional linear system with skew-adjoint generator," *Mathematics of Control, Signals, and Systems*, vol. 26, no. 3, pp. 435–462, 2014.

[4] J. W. Helton, "Systems with infinite-dimensional state space: the hilbert space approach," *Proceedings of the IEEE*, vol. 64, no. 1, pp. 145–160, 1976.

[5] K. Ramdani, M. Tucsnak, and G. Weiss, "Recovering the initial state of an infinite-dimensional system using observers," *Automatica*, vol. 46, no. 10, pp. 1616–1625, 2010.

[6] C.-Z. Xu, P. Ligarius, and J.-P. Gauthier, "An observer for infinite-dimensional dissipative bilinear systems," *Computers & Mathematics with Applications*, vol. 29, no. 7, pp. 13–21, 1995.

[7] H. Bounit and H. Hammouri, "Observers for infinite dimensional bilin ear, systems," *European journal of control*, vol. 3, no. 4, pp. 325–339, 1997.