EPFL

Geometric Computing Laboratory

Inverse Geometric Propulsion

A thesis submitted to the
School of Computer and Communication Sciences
of the École Polytechnique Fédérale de Lausanne
for the degree of

Master of Science

by

Orfeas Liossatos

Supervised by

Quentin Becker
School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne

Oliver Gross
Center for Visual Computing, Computer Science and Engineering
University of California, San Diego

Mark Pauly
School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne

Dated

June 27, 2025

Acknowledgements

For being stellar supervisors, Oliver Gross, Quentin Becker, Mark Pauly. For proof-reading, my friends Nuno Ricardo Cordeiro Serafim, Charlie Park, Camille Muzzetto.

Abstract

Controlling shape-changing bodies in zero-gravity is complex but critical for tasks like space robotics or animation. We extend a motion planning algorithm for shape-changing bodies to the control of non-zero momentum. By formulating physical motion as a energy-minimising trajectory on a configuration manifold, we enable the discovery of non-trivial, physically valid motion paths that combine shape change with momentum control. The use of a stable variational integrator for the forward simulation leads to an efficient optimisation algorithm. We validate our method on abstract mechanical systems such as rigid bodies and a hinged flap with thrusters. This approach can inform motion planning in space robotics, reinforcement learning, and procedural animation.[1]

[1]Project code is available at `https://github.com/orfeasliossatos/master-project`

Contents

Chapter 1

Introduction

Consider a falling cat, a platform diver, or an astronaut. In a reference frame moving with their trajectory, that is, where external forces are negligible, they are able to reorient themselves purely through internal shape changes. In this section we discuss how this is possible by putting ourselves in the boots of an astronaut, contrasting the Newtonian and Lagrangian perspectives on the dynamics of shape change. Next we discuss the challenges of controlling a shape-changing body via external forces such as the thrusters on a jetpack, before concluding with an overview of the project itself.

1.1 Turning around in space

In the microgravity of space, where there is nothing to push against, the law of conservation of momentum governs the motion of the astronaut. If the astronaut swings their arms to the left, their whole body swings to the right, and vice-versa. As a result, the astronaut cannot shift their centre of mass by moving their arms around. However, through a coordinated sequence of body movements, it is possible to redistribute angular momentum throughout the body, resulting in a total reorientation. For instance, an astronaut at rest can turn around by repeatedly sweeping their arms in a circular motion parallel to the ground. From a Newtonian perspective, muscles exert equal and opposite forces at their points of attachment, which allows one part of the body to rotate one direction while another rotates another direction, ensuring the constancy of angular momentum. However, since the opening and closing motions of the circle are not symmetric, the arms experience net torque during the motion, and the rest of the body experiences the opposite torque because angular momentum is conserved. So when the circular motion is closed and the arms return to their initial position, the astronaut has reoriented themself. This principle can be felt on Earth using a swivel chair and a heavy book.

This point-wise perspective is instructive, and readily leads to a proper computer simulation by integration of the second order forces. However, deriving the internal forces acting pointwise on the body is a potentially non-trivial and tedious exercise, depending on the complexity of the system. Complementing the Newtonian point of view is the following Lagrangian perspective on the dynamics, which expresses equivalent equations of motion in purely geometric terms. Think of the changing shape of the astronaut as a curve in a space of astronaut shapes. Then the actual configuration of the astronaut is given by rigidly transforming the shape onto some position in space. So the total configuration space naturally decomposes into shape and rigid transformation. Euler's least action principle then states that when the end points of a configuration curve are fixed, the motion achieved by a shape changing body is a curve that locally minimizes the kinetic energy of the body. Now suppose that the astronaut makes some periodic motion with their body, like drawing a circle with their arms. This constitutes a closed curve in shape space. So how does this result in a reorientation of the body? The idea is that the configuration space is intrinsically curved, and so a closed curve in shape space, when *lifted* to a curve on the total configuration space, can exhibit a so-called *geometric phase*.

By way of analogy, consider a closed curve on a sphere. Lifting the curve to the sphere's tangent bundle means selecting smoothly varying vectors in the tangent spaces traversed by the curve. Assuming the vectors are parallel, in the sense that that their derivative along the curve is zero, then the lifted curve itself may not be closed due to the curvature of the sphere. To feel this intuitively, one can stick their arm out, thumb pointing up. Then draw a large circle with their arm while keeping their wrist straight. On completing the circle, the thumb will point in some other direction. So the curve in thumb-space is not closed, and the difference between the original thumb direction and the new thumb direction is called the geometric phase.

We adopt this geometric perspective because it elegantly leads to explicit equations that are convenient for the computational simulation of the motion of shape-changing bodies. Another advantage of this perspective is that we need not explicitly state how the shape change comes about, we need only describe which shapes are possible at all. Finally, the geometric perspective is entirely equivalent to the Newtonian perspective, and we will show how one leads to the other through Euler's principle of least action and a Riemannian version of Noether's theorem.

1.2 Propulsion of shape-changing control systems

So far we have only considered systems with constant momentum. In this project, we break this assumption and aim to directly control the momentum of a shape-changing body via external forces. We take inspiration from dynamical systems that simultaneously leverage forces and shape change for motion. Consider for example the gimballed rocket engines of the Saturn V rocket and Space Shuttle. The direction of thrust can be turned away from the centre of mass of the rocket, which generates a torque about the centre of mass, allowing the the rocket to be controlled. Gimballed rockets are highly unstable, so the simultaneous control of thrust and shape change must be precisely understood for correct motion planning.

The same can be said for the Manned Maneuvring Unit, an astronaut jetpack designed for free flight in space [7]. NASA astronauts were once tasked with recovering a pair of satellites that did not reach their proper orbit [5]. The astronauts had to approach the satellites, grab them, slow their tumbling, and fly them back to the Space Shuttle. Consider again the control problem at hand. Although the jetpack features multi-directional thrusters, the astronaut can purposely change the direction of thrust by rotating their arms, shifting the orientation of the whole jetpack-astronaut system.

Thus, controlling the flight of a shape-changing body remains an important and challenging task. For instance, there is interest in piloting robotic arms for the purpose of repairing satellites and collecting space debris [1], which calls for efficient algorithms and a framework which allows us to study the interplay of forces and shape changes necessary for the motion planning of such systems. To this end, we extend the inverse design framework of Becker et al. to the control of momentum and forces. Once again, the abstract framework grants us the power of generality. In order to check our intuition, and gain a complete understanding of the control algorithm, we focus on highly simplified and abstract dynamical systems, with few or no moving parts, such as a totally rigid body and a single hinged flap.

1.3 Project overview

The purpose of this project is to extend the framework of Becker et al. to the simultaneous control of momentum and shape change. In chapter 2, we cover the necessary elements of *geometric mechanics* to describe the continuous motion of a shape-changing body in terms of an equivalent action principle. In chapter 3 we discuss the *inverse design problem* of motion planning with momentum and shape change. In chapter 4, we discretize the equations of motion and lay out the optimisation algorithm. In chapter 5 we apply the method to a few abstract dynamical systems in computer simulation. In chapter 6 we discuss the power and limitations of this approach.

Chapter 2

Background

We begin by describing the *geometric mechanics* of the space where the dynamics play out [9].

2.1 Rigid body mechanics

To describe how a shape is positioned in space, and how it is moving, we use the special Euclidean group SE(3), the group of rigid body transformations.

Lie group A rigid body transformation $g \in \mathrm{SE}(3)$ may be represented as a pair $g = (A, b)$, where $A \in \mathrm{SO}(3)$ is a rotation matrix and $b \in \mathbb{R}^3$ is a vector. These transformations act on $\mathbb{R}^3$ via

$$g : \mathbb{R}^3 \to \mathbb{R}^3, \quad x \mapsto Ax + b,$$

applying a rotation followed by a translation. Being both a group and a smooth manifold, SE(3) is a Lie group. In the context of rigid body mechanics, moving an object rigidly with g induces a change of frame of reference from the object's point of view such that a fixed point $x_{\text{world}} \in \mathbb{R}^3$ in the world frame located at $x_{\text{body}} \in \mathbb{R}^3$ in the initial body frame would be seen as moving to $g^{-1} \cdot x_{\text{body}}$ in the final body frame.

Lie algebra An infinitesimal rigid body transformation $Y \in \mathfrak{se}(3) := T_{Id}\,\mathrm{SE}(3)$ may be represented as a pair $Y = (\omega, v)$, where $\omega \in \mathbb{R}^3$ specifies the skew-symmetric matrix $[\omega]_\times$ corresponding to the cross product $\omega \times \cdot$, and $v \in \mathbb{R}^3$ is a vector. Every tangent vector in $T_g\,\mathrm{SE}(3)$ is the left-translation gY of some $Y \in \mathfrak{se}(3)$ by g (Figure 2.1).

Dual Lie algebra A body momentum $\mu \in \mathfrak{se}(3)^*$ may be represented as a pair $(l, p) \in \mathbb{R}^3 \times \mathbb{R}^3$, containing angular and linear momentum respectively. Its duality pairing with elements $Y \in \mathfrak{se}(3)$ is given by

$$\langle \mu | Y \rangle = \langle l, \omega \rangle + \langle p, v \rangle,$$

where the bilinear operator $\langle \cdot, \cdot \rangle$ is the standard inner product on $\mathbb{R}^3$. An infinitesimal momentum, or body force $f \in \mathfrak{se}(3)^*$ may be represented similarly.

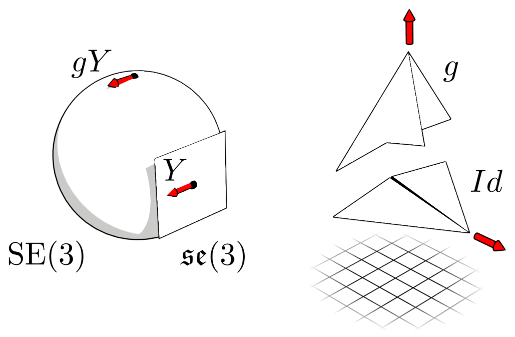


Figure 2.1: A depiction of the special Euclidean group.

Coadjoint action Given a rigid transformation $g \in \mathrm{SE}(3)$, the group coadjoint action $\mathrm{Ad}_g^* : \mathfrak{se}(3)^* \to \mathfrak{se}(3)^*$ is given by

$$(l, p) \mapsto (Al + b \times p, Ap),$$

and it expresses the momentum or forces exerted in a reference frame given by g to the world frame (given by $Id \in \mathrm{SE}(3)$). We use the notation $\mu_{\mathrm{world}} = \mathrm{Ad}_g^* \, \mu_{\mathrm{body}}$ and $f_{\mathrm{world}} = \mathrm{Ad}_g^* \, f_{\mathrm{body}}$.

2.2 Configuration manifold

The configuration space $\mathcal{M}$ is a Riemannian manifold of shapes in correspondence in $\mathbb{R}^3$. This contains all the possible states of the dynamical system. That is, all shapes in all positions.

Shape space Two bodies $\gamma_1, \gamma_2 \in \mathcal{M}$ have the same *shape* if there exists a transformation $g \in \mathrm{SE}(3)$ such that $\gamma_2 = g \cdot \gamma_1$ pointwise. This defines an equivalence relation on $\mathcal{M}$ with equivalence classes

$$[\gamma] = \{g \cdot \gamma : g \in \mathrm{SE}(3)\} \subset \mathcal{M}$$

for any $\gamma \in \mathcal{M}$. The quotient projection $\pi : \mathcal{M} \to \mathcal{M}/\,\mathrm{SE}(3)$ defines the *shape space* $\mathcal{S} := \mathcal{M}/\,\mathrm{SE}(3)$.

Product manifold The configuration space $\mathcal{M}$ is isomorphic to $\mathcal{S} \times \mathrm{SE}(3)$, meaning that we may refer to configurations by a pair (s, g) containing a shape $s \in \mathcal{S}$ and a rigid transformation $g \in \mathrm{SE}(3)$ that places the shape in the world frame by the action $\gamma = g \cdot s$. Given a sequence of shapes $s : [0, T] \to \mathcal{S}$, we will be interested in certain choices of *lifts* $s \mapsto (s, g)$ to the full configuration space. Not all lifts are physically meaningful, but we will show how to find lifts that obey physical law, specifically Euler's Least Action principle.

Riemannian metric The metric $\langle \cdot, \cdot \rangle_{\mathcal{M}}$ is used to model the kinetic energy of the body in a vacuum. The metric therefore accounts for the mass distribution of the body. We choose an SE(3)-invariant metric, meaning $\langle gZ_1, gZ_2 \rangle_{\mathcal{M}} = \langle Z_1, Z_2 \rangle_{\mathcal{M}}$ for any $g \in$ SE(3) and $Z_1, Z_2 \in T\mathcal{M}$, so that the energy of a motion trajectory is invariant under global rigid body motions. This means the Lagrangian of our system has symmetries, which ultimately gives rise to known physical laws such as the *conservation of momentum* for rigid bodies via Noether's theorem (Theorem 1).

2.3 Physics of shape change

We will summarise the equations of motion for shape-changing bodies in a vacuum.

Principle of Stationary Action We view motion as a time-varying configuration $\gamma : [0, T] \to \mathcal{M}$. In the absence of forces, the Euler-Lagrange principle of stationary action states that physical motions between fixed start and end configurations are the stationary points of the energy functional

$$\mathcal{E}(\gamma) = \frac{1}{2} \int_0^T \langle \gamma', \gamma' \rangle_{\mathcal{M}} \mathrm{d}t, \tag{2.1}$$

under variations of γ, where $\gamma' : [0, T] \to T\mathcal{M}$ is the tangent curve to γ, and the metric on $\mathcal{M}$ computes the kinetic energy in a vacuum. A *variation* is an assignment of a tangent vector to each point along γ, so it is the restriction to points in γ of a vector field $\delta\gamma \in \Gamma T\mathcal{M}$, the collection of all vector fields.

Vertical distribution Motion planning requires knowing which rigid body transformations will result from a body undergoing shape change. So we assume that the shapes $t \mapsto s(t)$ of the time-varying configuration are known, and we look for stationary curves under variations of the sequence of rigid body transformations $t \mapsto g(t)$. To construct a notion of variation purely in the direction of an infinitesimal rigid body motion $Y \in \mathfrak{se}(3)$, consider some curve $\varepsilon \mapsto h(\varepsilon) \in$ SE(3) such that $h'(0) = Y$ and $h(0) = Id$. Then we can build a so-called *vertical* vector field $\hat{Y} \in \Gamma T\mathcal{M}$ on the full configuration space with vectors

$$\hat{Y}_\gamma = \left. \frac{d}{d\varepsilon} \right|_{\varepsilon=0} h(\varepsilon) \cdot \gamma,$$

for all $\gamma \in \mathcal{M}$. This shows that vertical vector fields $\hat{Y} \in \Gamma T\mathcal{M}$ and infinitesimal rigid body motions $Y \in \mathfrak{se}(3)$ are in one-to-one correspondence. The collection of all such fields is called the *vertical distribution* $V \subset T\mathcal{M}$. A *vertical variation* is then nothing other than a curve in V.

Killing vector field We will now show that a vertical vector field $\hat{Y} \in \Gamma T\mathcal{M}$ is also a *Killing vector field*. Geometrically, this means that curvature does not change along $\hat{Y}$. Physically, it means that energy is invariant to infinitesimal changes of position and orientation. Recall that for any $g \in$ SE(3) and $Z_1, Z_2 \in T\mathcal{M}$ we have

$$\langle gZ_1, gZ_2 \rangle_{\mathcal{M}} = \langle Z_1, Z_2 \rangle_{\mathcal{M}}.$$

This can also be seen as the pullback of g on the bilinear form $\langle \cdot, \cdot \rangle_{\mathcal{M}}$, written

$$(g^* \langle \cdot, \cdot \rangle_{\mathcal{M}})(Z_1, Z_2) = \langle \cdot, \cdot \rangle_{\mathcal{M}}(Z_1, Z_2),$$

because g is an isometry, so g^* is an identity map. Now pick some vertical vector field $\hat{Y} \in \Gamma T\mathcal{M}$ built from the family $\varepsilon \mapsto h(\varepsilon) \in \mathrm{SE}(3)$ with $h'(0) = Y$. Then the Lie derivative of the metric $\langle \cdot, \cdot \rangle_{\mathcal{M}}$ along $\hat{Y}$ is

$$\mathcal{L}_{\hat{Y}} \langle \cdot, \cdot \rangle_{\mathcal{M}} := \left. \frac{\mathrm{d}}{\mathrm{d}\varepsilon} \right|_{\epsilon=0} h(\varepsilon)^* \langle \cdot, \cdot \rangle = 0$$

because $h(\varepsilon)^*$ is the identity for every ε. This makes $\hat{Y}$ a Killing vector field by definition. Equivalently, in terms of the Levi-Civita connection we have

$$\langle \nabla_{Z_1} \hat{Y}, Z_2 \rangle_{\mathcal{M}} + \langle Z_1, \nabla_{Z_2} \hat{Y} \rangle_{\mathcal{M}} = 0$$

for any $Z_1, Z_2 \in T\mathcal{M}$ [6]. Now if we let $Z_1 = Z_2$, the following lemma will prove useful.

Lemma 1. *If $X \in \Gamma T\mathcal{M}$ is a Killing vector field and $Z \in T\mathcal{M}$, then*

$$\langle \nabla_Z X, Z \rangle = 0.$$

Proof. By rearranging the definition of a Killing vector field, we have

$$\langle \nabla_Z X, Z \rangle_{\mathcal{M}} = -\langle Z, \nabla_Z X \rangle_{\mathcal{M}}.$$

Now notice that the Riemannian metric is symmetric, so

$$\langle \nabla_Z X, Z \rangle_{\mathcal{M}} = -\langle \nabla_Z X, Z \rangle_{\mathcal{M}},$$

which is only possible for a real number if it is zero. □

Dynamics on the tangent bundle Now we derive the equations of motion on the tangent bundle $T\mathcal{M}$. For a variation $t \mapsto \delta\gamma(t)$ with fixed endpoints $\delta\gamma(0) = \delta\gamma(T) = 0$, integration by parts on a manifold yields

$$\delta\mathcal{E}(\gamma) = \frac{1}{2} \int_0^T \delta \langle \gamma', \gamma' \rangle \mathrm{d}t = \cancelto{0}{[\langle \delta\gamma, \gamma' \rangle]_0^T} - \int_0^T \langle \delta\gamma, \gamma'' \rangle \mathrm{d}t. \tag{2.2}$$

Now suppose we consider only vertical (rigid body) variations, so that $\delta\gamma \in \Gamma V$. The following theorem, a Riemannian variant of Noether's theorem, establishes that stationary motion trajectories have a conserved quantity for each symmetry of the Lagrangian Equation 2.1

Theorem 1 (Riemannian Noether's Theorem). *A curve $t \mapsto \gamma(t)$ is a stationary point of $\mathcal{E}(\gamma)$ under vertical variations with fixed endpoints if and only if for all $\delta\gamma \in V$,*

$$\langle \delta\gamma, \gamma' \rangle_{\mathcal{M}}$$

is constant as a function of t.

Proof. For the first direction, choose a vertical variation $\delta\gamma \in V$. Since $\delta\gamma$ is vertical, it is also Killing, so by Lemma 1, we have

$$\langle \delta\gamma, \gamma' \rangle'_{\mathcal{M}} = \langle \nabla_{\gamma'} \delta\gamma, \gamma' \rangle_{\mathcal{M}} + \langle \delta\gamma, \gamma'' \rangle_{\mathcal{M}} = \langle \delta\gamma, \gamma'' \rangle_{\mathcal{M}}. \tag{2.3}$$

Then by taking arbitrary sub-intervals of Equation 2.2, the criticality assumption $\delta\mathcal{E}(\gamma) = 0$ shows that

$$\langle \delta\gamma, \gamma'' \rangle_{\mathcal{M}}(t) = 0,$$

for all $t \in [0, T]$, and thus $\langle \delta\gamma, \gamma' \rangle_{\mathcal{M}}$ is constant as a function of time.

For the converse, Equation 2.3 shows that $\langle \delta\gamma, \gamma'' \rangle_{\mathcal{M}} = 0$, so we immediately have $\delta\mathcal{E}(\gamma) = 0$.

$\square$

Conservation of momentum Theorem 1 can be interpreted as the conservation of momentum. Suppose that the infinitesimal rigid body transformation $Y \in \mathfrak{se}(3)$ represented by (ω, v) generates the vertical vector field $\hat{Y}$, and that the body $\gamma(t)$ has mass density ρ. Then by expanding the metric on $\mathcal{M}$ we recover the pairing of linear and angular momentum with Y, since

$$\begin{aligned}
\langle \hat{Y}_{\gamma(t)}, \gamma'(t) \rangle_{\mathcal{M}} &= \langle \omega \times \gamma(t) + v, \gamma'(t) \rangle_{\mathcal{M}} \\
&= \left\langle \int_{\gamma(t)} x \times x' \mathrm{d}\rho(x), \omega \right\rangle + \left\langle \int_{\gamma(t)} x' \mathrm{d}\rho(x), v \right\rangle \\
&:= \langle \mu_{\gamma'(t)} | Y \rangle,
\end{aligned}$$

where $\langle \cdot | \cdot \rangle$ is the duality pairing on $\mathfrak{se}(3)$. From considering motions $Y = (\omega, 0)$ we conclude that the angular momentum $\int_{\gamma(t)} x \times x' \mathrm{d}\rho(x)$ is conserved, while considering only $Y = (0, v)$ yields the conservation of linear momentum $\int_{\gamma(t)} x' \mathrm{d}\rho(x)$. Since ω and v are vectors in $\mathbb{R}^3$, there are a total of 6 conserved quantities. This motivates the definition of the *momentum map*

$$\mu : T\mathcal{M} \to \mathfrak{se}(3)^*, \quad Z \mapsto \mu_Z,$$

returning a *geometric momentum* $\langle \mu_Z | Y \rangle := \langle \hat{Y}, Z \rangle_{\mathcal{M}}$. Then combining Theorem 1 with the identification $\mathfrak{se}(3)^* \cong \mathbb{R}^3 \times \mathbb{R}^3$ implies the conservation of momentum

$$\mu_{\gamma'(t)} = \mu_0 \in \mathbb{R}^3 \times \mathbb{R}^3,$$

which, as a constraint in six equations and six unknowns (the components of the future rigid body transformation) is nicely enforced in a computer by solving the differential equation, which is first order.

External forces So far, we have described the motion of a body in the absence of external forces. However, we are in particular interested in controlling the momentum of a shape-changing body for motion planning. Supposing the body is initially at rest ($\mu_0 = 0$), with time-evolving momentum $t \mapsto \mu_{\text{world}}(t)$. In that case, we instead write

$$\mu_{\gamma'(t)} = \mu_{\text{world}}(t).$$

Then, according to Newton's second law, the body must be pushed by a continuously evolving world force $t \mapsto f_{\text{world}}(t)$, which is the rate of change of momentum

$$\frac{\mathrm{d}\mu_{\gamma'(t)}}{\mathrm{d}t} = f_{\text{world}}(t).$$

Chapter 3

Inverse Design

Previously, we described a procedure for lifting a prescribed shape sequence $t \mapsto s(t)$ to a full configuration sequence $t \mapsto \gamma(t)$ by finding a physically consistent sequence of rigid body transformations $t \mapsto g(t)$ with time-dependent momentum $t \mapsto \mu_{\text{world}}(t)$. In this chapter we formalize the *inverse geometric propulsion* problem. That is, searching for shapes and momenta that minimize an objective function $\mathcal{J}$ of the form

$$\mathcal{J} : (\gamma : [0,T] \to \mathcal{M}) \to \mathbb{R}_{\geq 0},$$

while maintaining physical consistency as the constrained optimisation problem,

$$\underset{\substack{s:[0,T]\to\mathcal{S} \\ \mu_{\text{world}}:[0,T]\to\mathfrak{se}(3)^*}}{\arg\min} \quad \mathcal{J}(\gamma) \quad \text{s.t. } \mu_{\gamma'(t)} = \mu_{\text{world}}(t), \quad \forall t \in [0,T]. \tag{3.1}$$

This formulation does not reference forces directly, and this is by design. In the following, we discuss the kinds of motion objective functions we use in our experiments, and various ways of parametrising the shape and momentum sequences. In particular, force-based parametrisations of the momentum allows regularising e.g., fuel consumption, more naturally.

3.1 Motion objectives

A basic motion target is the requirement that the body reaches a position and orientation by the end of the motion. This *positioning* target is expressed by the objective function

$$\mathcal{J}_{\text{pos}}(t \mapsto g(t)) = \frac{1}{2}\|g^* - g(T)\|^2_{\text{SE}(3)},$$

where $g^* \in \text{SE}(3)$ is the target position, and $\|\cdot\|_{\text{SE}(3)}$ is a norm on SE(3). If a sequence of targets (g^*_k) are to be reached at a sequence of prescribed times (t_k), then the *checkpoints* objective function expresses it via

$$\mathcal{J}_{\text{cpt}}(t \mapsto g(t)) = \sum_k \frac{1}{2}\|g(t_k) - g^*_k\|^2_{\text{SE}(3)}.$$

An example motion path optimising for the checkpoints objective can be seen in Figure 5.1 and others. Note that there exist more flexible objective functions where the time-points (t_k) are unspecified, constructed for instance with chamfer distances, or smooth versions thereof [3].

Shape objectives One may also specify a number of shape objectives, for example to promote motion that avoids collision with obstacles. In our experiments, we focus on position-based objectives and highlight effects related to our contributions.

3.2 Design variables

We may wish to parametrise semantically meaningful regions of the shapes and momenta with higher-level design variables, such as body angles and forces. We use gradient-based methods for optimisation and require the map from parameters to the shapes and momenta to be differentiable.

3.2.1 Shape parametrisation

In physical scenarios, we may not have control over each individual point of a shape, and instead control groups of points through *shape parameters*. For example, we may wish to control the joint angles of a robotic arm, or the muscles of a soft-bodied creature. Therefore, we conceptualise a parametrisation as a differentiable map $\varphi : \mathcal{P} \to \mathcal{S}$ from a parameter space $\mathcal{P} \subset \mathbb{R}^p$ to the shape space $\mathcal{S}$, where the image of φ is understood as the set of admissible shapes.

Dihedral angles In our experiments we focus on the simple mechanical system of a hinge Figure 5.5. The hinge is parametrised by a single dihedral angle that opens and closes the hinge. The dihedral angle then controls the rotation of a mesh around the axis of the hinge, where an angle of 0 degrees represents a fully opened hinge and angles $\pi/2$ and $-\pi/2$ represent fully closed hinges in either direction.

3.2.2 Momentum parametrisation

In section 4.2, we show how to control the world momentum $t \mapsto \mu_{\mathrm{world}}(t) \in \mathbb{R}^3 \times \mathbb{R}^3$of a shape changing body, in order to achieve some motion objective. In the following we show how the world momentum itself can be computed from various other quantities summarised in Table 3.1.

Body momentum We may directly design the momentum as seen from the moving body frame via the group co-adjoint action

$$\mu_{\mathrm{world}}(t) = \mathrm{Ad}^*_{g(t)}\, \mu_{\mathrm{body}}(t)$$

World forces By Newton's second law, we parametrise the momenta by world forces

$$\mu_{\mathrm{world}}(t) = \int_0^t f_{\mathrm{world}}(\tau) \mathrm{d}\tau.$$

Body and directional forces In physical scenarios it makes more sense to control *body forces*, which are intended to be applied to the centre of mass of the shape over time and are viewed in the fixed body frame. If in the fixed body frame, the shape's centre of mass is located at x_{com}, then a body placed at $g \in \mathrm{SE}(3)$ experiences world-frame forces,

$$f_{\mathrm{world}} = \mathrm{Ad}^*_{g \cdot g_{\mathrm{com}}}(f_{\mathrm{body}}),$$

where $g_{\text{com}} = (I_3, x_{\text{com}}) \in \text{SE}(3)$ translates the body forces to centre of mass. Therefore, to simplify the mathematics, we assume that shapes are always centred on their centre of mass, so that $x_{\text{com}} = 0$ and

$$f_{\text{world}} = \text{Ad}_g^*(f_{\text{body}}).$$

For physical systems such as rockets, we may only have control of the magnitude of the forces, but not their direction, which is given by a normalised vector f_{fixed}. In that case we have

$$f_{\text{world}} = \text{Ad}_g^*(\|f\| f_{\text{fixed}}).$$

Offset and pinned forces If a force instead pushes at some fixed offset from the body's origin, where the offset is given by a rigid body transformation $h \in \text{SE}(3)$, then we have

$$f_{\text{world}} = \text{Ad}_{gh}^*(f_{\text{offset}}).$$

An advantage of offsetting forces is the ability to optimize multiple forces simultaneously at different offsets. Pushing this idea further is the ability to pin a force onto the shape change itself, such as the force aimed by a gimballed rocket nozzle. Mathematically, we define a differentiable map $h : \mathcal{P} \to \text{SE}(3)$ from shape parameters to offsets such that a body (s, g) experiences forces

$$f_{\text{world}} = \text{Ad}_{gh(s)}^*(f_{\text{pinned}})$$

in the world frame.

Quantity	Relation to $\mu_{\text{world}}(t)$
Body momentum μ_{body}	$\text{Ad}_{g(t)}^* \, \mu_{\text{body(t)}}$
World forces f_{world}	$\int_0^t f_{\text{world}}(\tau)\text{d}\tau$
Body forces f_{body}	$\int_0^t \text{Ad}_{g(\tau)}^* \, f_{\text{body}}(\tau)\text{d}\tau$
Thrust $\|f\|$	$\int_0^t \|f\|(\tau) f_{\text{fixed}}\text{d}\tau$
Pinned forces f_{pinned}	$\int_0^t \text{Ad}_{g(\tau)h(s(\tau))}^* \, f_{\text{pinned}}(\tau)\text{d}\tau$

Table 3.1: Table of world momentum parametrisations.

3.3 Regularisation

If it is desired that the forces $t \mapsto f(t)$ are minimal, expressing the need for the economical use of fuel, then one can simultaneously minimise a L1-regularisation function such as

$$\mathcal{R}(t \mapsto f(t)) = \int_0^T \|f(t)\|_1 \text{d}t,$$

in order to encourage sparse activation of forces. The regularisation terms and motion objective functions may be linearly combined with weights $\lambda > 0$, but this creates a trade-off between objectives, as we explore in Figure 5.3.

Chapter 4

Method overview

We describe an algorithmic approach to the *inverse geometric propulsion* problem. To achieve this, we propose an extension of the methods presented by Becker et al., for which we provide a brief overview.

4.1 Discretisation

The configuration space $\mathcal{M}$ of bodies is discretised as $M := (\mathbb{R}^3)^N$, where a body is represented by an ordered list of vertices

$$(\gamma_1, \dots, \gamma_N) \in M.$$

Therefore, the discrete shape space is the corresponding quotient $S := M/\operatorname{SE}(3)$, and we study time-discrete sequences of bodies $(\gamma^1, \dots, \gamma^T) \in M^T$ with fixed-length time-steps $\Delta t = 1$.

Discrete variational energy The discrete version of Equation 2.1 is

$$E(\gamma) = \frac{1}{2} \sum_{t=1}^{T-1} \langle \Delta\gamma^{t+1}, \Delta\gamma^{t+1} \rangle_M,$$

where $\Delta\gamma^{t+1} := \gamma^{t+1} - \gamma^t$ and $\langle \cdot, \cdot \rangle_M$ is an SE(3)-invariant Riemannian metric on M. While we could consider both isotropic and anisotropic dissipation metrics as in Becker et al., we focus on purely inertial settings (the former case), so the metric models kinetic energy and is given by

$$\langle u, v \rangle_M = \sum_{j=1}^{N} m_j \langle u_j, v_j \rangle_{\mathbb{R}^3},$$

with masses m_j at each vertex. In our experiments involving surface meshes, the mass at a vertex is proportional to the area of its Voronoi cell, and thus may in principle evolve over time if the total surface area of the mesh changes. However, our experiments only involve area-preserving transformations of the mesh (like folding about an axis), and thus the masses are computed once and kept constant.

Integrating shapes to motion Given a sequence of input shapes $(s^1, \dots, s^T)$ and world momenta $(\mu^2, \dots, \mu^T)$ the motion trajectory of the body is uniquely determined up to a global rigid body transformation. So fixing the first rigid body transformation $g^1 = Id \in \mathrm{SE}(3)$ determines the rest by solving the physical consistency constraint at each time step. Specifically, the choice of rigid body transformations $(g^2, \dots, g^T) \in \mathrm{SE}(3)^{T-1}$ is made so that the discrete momentum

$$\mu(\gamma^{t-1}, \gamma^t) := \begin{pmatrix} -\sum_{j=1}^N m_j(\gamma_j^{t-1} \times \gamma_j^t) \\ -\sum_{j=1}^N m_j \Delta\gamma^t \end{pmatrix}$$

matches the designed μ^t. This is enforced computationally by iteratively finding zeros of the *residual momentum* constraint function

$$\varphi^t : \mathrm{SE}(3) \to \mathbb{R}^3 \times \mathbb{R}^3, \quad g^t \mapsto \mu(\gamma^{t-1}, g^t \cdot s^t) - \mu^t, \tag{4.1}$$

for $t = 2, \dots, T$. Algorithm 1 describes a *variational integrator*, which is known to have nice properties such as long-term energy preservation [10].

Algorithm 1 IntegrateTrajectory

```
1: Input: shapes $s \in S^T$, momenta $\mu^2, \dots, \mu^T \in \mathbb{R}^6$
2: Output: physical lift $\gamma \in M^T$.
3: $\gamma^1 \leftarrow Id(s^1)$
4: for $t = 2, \dots, n$ do
5:     $g^t \leftarrow$ solve $\mu(\gamma^{t-1}, g^t \cdot s^t) - \mu^t = 0$        ▷ Equation 4.1
6:     $\gamma^t \leftarrow g^t \cdot s^t$
7: end for
```

We solve Equation 4.1 by feeding the Jacobian of φ^t to MINPACK's HYBRD algorithm [11]. Although the rigid transformations are SE(3) elements, in practice we allow the root solver to return elements that do not strictly live on SE(3), and instead simultaneously minimize the local defining function for SE(3)

$$\mathbb{R}^{12} \cong \mathbb{R}^{3\times3} \times \mathbb{R}^3 \to \mathbb{R}, \quad (A, b) \mapsto \|A^\top A - I\|.$$

and further project the result onto SE(3). For this reason, we compute Euclidean derivatives rather than Lie derivatives of φ^t with respect to the rigid body transformations. In our code, we build the partial Jacobian $\partial\varphi^t/\partial g^t$ as a matrix of directional derivatives where each row is given by

$$\left(\frac{\partial\varphi^t}{\partial g^t}\right)_i = D\varphi^t(g^t)[e_i] = \begin{pmatrix} -\sum_{j=1}^N m_j(\gamma_j^{t-1} \times D\gamma_j^t(g^t)[e_i]) \\ -\frac{1}{2}\sum_{j=1}^N m_j D\gamma_j^t(g^t)[e_i] \end{pmatrix}^\top - D\mu^t(g^t)[e_i] \tag{4.2}$$

for basis vectors $e_i \in \mathbb{R}^{12}$. Note that some parametrisations of μ^t depend on g^t, so $D\mu^t(g^t)[e_i]$ is in general non-zero, and its computation is addressed in section 4.3. To compute $D\gamma_j^t(g^t)[e_i]$, note that the Euclidean directional derivative of $\gamma(g) = g \cdot s = As + b$ is given by

$$D\gamma(g)[g'] = A'\gamma(g) + b'.$$

For the full control algorithm, we also require partial Jacobians $\partial\varphi^{t+1}/\partial g^t$ for $t = 2 \dots T-1$, which is computed similarly.

4.2 Optimisation algorithm

To solve the optimisation problem in Equation 3.1, we compute the gradient of the objective function with respect to the design variables, and feed the gradients to the L-BFGS optimisation algorithm [4]. The following theorem shows how to compute gradients with respect to any design variable related to momentum, denoted by ν.

Theorem 2. *Given a design variable ν, independent of the shapes $s \in S^T$, the partial differential of a position-based objective function $J(g_2, \dots, g_T)$ with respect to ν is*

$$\frac{\partial J}{\partial \nu} = -\sum_{t=2}^{T} w_t^\top \frac{\partial \mu^t}{\partial \nu},$$

where $g_2, \dots, g^T$ are the zeros of $\varphi^2, \dots, \varphi^T$, and the adjoint vectors w_t are solutions to the recursive linear equations

$$-\frac{\partial \varphi^t}{\partial g^t}^\top w_t = \frac{\partial \varphi^{t+1}}{\partial g^t} w_{t+1} + \frac{\partial J}{\partial g^t}, \tag{4.3}$$

with $w_{T+1} = 0$ as a base of the recursion.

Proof. First, consider the total differential

$$\frac{\mathrm{d}\varphi^t}{\mathrm{d}\nu} = \frac{\partial \varphi^t}{\partial g^t}\frac{\partial g_t}{\partial \nu} + \frac{\partial \varphi^t}{\partial g^{t-1}}\frac{\partial g^{t-1}}{\partial \nu} - \frac{\partial \mu^t}{\partial \nu}.$$

Since g^t are assumed to be the zeros of φ^t, and this remains true for any variation of ν, we have $\frac{\mathrm{d}\varphi^t}{\mathrm{d}\nu} = 0, \forall t$, and so we may deduce the differentials of the positions

$$\frac{\partial g_t}{\partial \nu} = \left(\frac{\partial \varphi^t}{\partial g^t}\right)^{-1}\left(\frac{\partial \mu^t}{\partial \nu} - \frac{\partial \varphi^t}{\partial g^{t-1}}\frac{\partial g^{t-1}}{\partial \nu}\right),$$

while the differential of g_2 simplifies to

$$\frac{\partial g_2}{\partial \nu} = \left(\frac{\partial \varphi^2}{\partial g^2}\right)^{-1}\frac{\partial \mu^2}{\partial \nu},$$

since g_1 is fixed. Next, we compute the differential of the objective. Expanding the first term we find

$$\frac{\partial J}{\partial \nu} = \frac{\partial J}{\partial g^T}\frac{\partial g^T}{\partial \nu} + \sum_{t=2}^{T-1}\frac{\partial J}{\partial g^t}\frac{\partial g^t}{\partial \nu} \tag{4.4}$$

$$= \frac{\partial J}{\partial g^T}\left(\frac{\partial \varphi^T}{\partial g^T}\right)^{-1}\frac{\partial \mu^T}{\partial \nu} - \frac{\partial J}{\partial g^T}\left(\frac{\partial \varphi^T}{\partial g^T}\right)^{-1}\frac{\partial \varphi^T}{\partial g^{T-1}}\frac{\partial g^{T-1}}{\partial \nu} + \sum_{t=2}^{T-1}\frac{\partial J}{\partial g^t}\frac{\partial}{\partial \nu}g^t, \tag{4.5}$$

which motivates use of the *adjoint vector w_T* in Equation 4.3. Plugging this back into Equation 4.5 we find

$$\frac{\partial J}{\partial \nu} = -w_T^\top \frac{\partial \mu^T}{\partial \nu} + w_T^\top \frac{\partial \varphi^T}{\partial g^{T-1}}\frac{\partial g^{T-1}}{\partial \nu} + \sum_{t=2}^{T-1}\frac{\partial J}{\partial g^t}\frac{\partial g^t}{\partial \nu} \tag{4.6}$$

$$= -w_T^\top \frac{\partial \mu^T}{\partial \nu} + \left(\frac{\partial \varphi^T}{\partial g^{T-1}}^\top w_T + \frac{\partial J}{\partial g^{T-1}}^\top\right)^\top \frac{\partial g^{T-1}}{\partial \nu} + \sum_{t=2}^{T-2}\frac{\partial J}{\partial g^t}\frac{\partial g^t}{\partial \nu}, \tag{4.7}$$

which exhibits a recursive structure, motivating the use of the next adjoint vector w_{T-1} in Equation 4.7, yielding

$$\begin{aligned}\frac{\partial J}{\partial \nu} &= -w_T^\top \frac{\partial \mu^T}{\partial \nu} - w_{T-1}^\top \left(\frac{\partial \mu^{T-1}}{\partial \nu} - \frac{\partial \varphi^{T-1}}{\partial g^{T-2}} \frac{\partial g^{T-2}}{\partial \nu} \right) + \sum_{t=1}^{T-2} \frac{\partial J}{\partial g^t} \frac{\partial g^t}{\partial \nu} \\ &= -\sum_{t=T-1}^{T} w_t^\top \frac{\partial \mu^t}{\partial \nu} + \left(\frac{\partial \varphi^{T-1}}{\partial g^{T-2}}^\top w_{T-1} + \frac{\partial J}{\partial g^{T-2}}^\top \right)^\top \frac{\partial g^{T-2}}{\partial \nu} + \sum_{t=1}^{T-3} \frac{\partial J}{\partial g^t} \frac{\partial g^t}{\partial \nu}.\end{aligned}$$

Repeating this process, we get

$$\frac{\partial J}{\partial \nu} = -\sum_{t=3}^{T} w_t^\top \frac{\partial \mu^t}{\partial \nu} + \left(\frac{\partial \mu^3}{\partial g^2}^\top w_3 + \frac{\partial J}{\partial g^2}^\top \right)^\top \frac{\partial g^2}{\partial \nu} = -\sum_{t=2}^{T} w_t^\top \frac{\partial \mu^t}{\partial \nu},$$

which concludes the proof. □

Thus we have the following optimisation algorithm for the design of world momenta. The algorithm must be adapted for higher level design variables using the chain rule, which we explore in section 4.3.

Algorithm 2 OptimizeMomentumSequence

```
 1: Input: objective $J : M^T \to \mathbb{R}_{\geq 0}$, initial guess $s \in S^T, \mu \in (\mathbb{R}^6)^{T-1}$
 2: Output: optimal shape and momentum sequences $s^* \in S^T, \mu^* \in (\mathbb{R}^6)^{T-1}$.
 3: while not Converged(J($\gamma$)) do
 4:     $\gamma \leftarrow$ IntegrateTrajectory($s, \mu$)                    ▷ Algorithm 1
 5:     $\mathbf{L}_T, \mathbf{r}_T \leftarrow -\frac{\partial \varphi^T}{\partial g^T}^\top, \frac{\partial J}{\partial g^T}$
 6:     $w_T \leftarrow$ Solve($\mathbf{L}_T w_T = \mathbf{r}_T$)                    ▷ Equation 4.3
 7:     $\frac{\mathrm{d}J}{\mathrm{d}s^T} \leftarrow -w_T^\top \frac{\partial \mu^T}{\partial s^T}$
 8:     $\frac{\mathrm{d}J}{\mathrm{d}\mu^T} \leftarrow -w_T^\top$
 9:     for $t = T-1, \dots, 2$ do
10:         $\mathbf{L}_t, \mathbf{r}_t \leftarrow -\frac{\partial \varphi^t}{\partial g^t}^\top, \frac{\partial \varphi^{t+1}}{\partial g^t} w_{t+1} + \frac{\partial J}{\partial g^t}$
11:         $w_t \leftarrow$ Solve($\mathbf{L}_t w_t = \mathbf{r}_t$)                ▷ Equation 4.3
12:         $\frac{\mathrm{d}J}{\mathrm{d}s^t} \leftarrow -w_t^\top \frac{\partial \mu^t}{\partial s^t} - w_{t+1}^\top \frac{\partial \mu^{t+1}}{\partial s^t}$
13:         $\frac{\mathrm{d}J}{\mathrm{d}\mu^t} \leftarrow -w_t^\top$
14:     end for
15:     $\mathbf{d} \leftarrow$ L-BFGS($s, \mu, \frac{\mathrm{d}J}{\mathrm{d}s}, \frac{\mathrm{d}J}{\mathrm{d}\mu}$)                 ▷ Descent direction
16:     $s, \mu \leftarrow$ Linesearch($J, s, \mu, \mathbf{d}$)
17: end while
```

4.3 Gradients with respect to design variables

Since the control of shapes in covered in detail in Becker et al., we focus on the control of design variables related to world momentum μ^t as per subsection 3.2.2. To recap, depending on the choice of parametrisation μ^{t+1} could depend on g^t. For instance, if μ^{t+1} is parametrised from body momenta via the formula $\mu^{t+1} = \mathrm{Ad}^*_{g^t} \mu^{t+1}_{\text{body}}$. Therefore, to build the Jacobian $\partial \varphi^{t+1} / \partial g^t$ in the algorithm, we compute the directional derivative of μ^{t+1} with respect to g^t along an arbitrary g'. Then, with the chain rule, we show how to deduce the gradient of J with respect to the chosen design variables. For these reasons, we provide these formulae in we summarise in Table 4.1.

Parametrisation	Gradient $\left(\frac{\partial J}{\partial \nu}\right)$	Directional Derivative $D\mu^{t+1}(g^t)[g']$
World momentum μ_t	$-w_t^\top$	0
Body momentum μ^t_{body}	$-w_t^\top[\text{Ad}^*_{g^{t-1}}]$	$(A'l + b \times A'p + b' \times Ap,\ A'p)$
World forcesf^t_{world}	$-\sum_{\tau=t}^T w_\tau^\top$	0
Body forces f^t_{body}	$-\sum_{\tau=t}^T w_\tau^\top[\text{Ad}^*_{g^{\tau-1}}]$	Same as for body momentum, with f^{t+1}_{body}
Directional thrust $\Vert f^t\Vert f^t_{\text{fixed}}$	$\dfrac{\partial J}{\partial f^t_{\text{body}}} \cdot f^t_{\text{fixed}}$	Same as above, with $f^{t+1}_{\text{body}} = \Vert f_{t+1}\Vert f^{t+1}_{\text{fixed}}$
Offset / pinned forces f^t_{offset}	$\dfrac{\partial J}{\partial f^t_{\text{body}}}[\text{Ad}^*_{h(s^{t-1})}]$	Same as above, with $f^{\text{body}}_{t+1} = \text{Ad}^*_{h(s^t)} f^{\text{offset}}_{t+1}$

Table 4.1: Gradients and directional derivatives for different momentum parametrisations.

Discrete world momentum The simplest case is the direct control of the world-frame momenta μ^t, for which Theorem 2 gives

$$\frac{\partial J}{\partial \mu^t} = -w_t^\top$$
$$D\mu^{t+1}(g^t)[g'] = 0.$$

Discrete body momentum If instead one controls the body momentum μ^t_{body} through the change-of-frame parametrisation

$$\mu^t = \text{Ad}^*_{g^{t-1}}(\mu^t_{\text{body}})$$

then we have

$$\frac{\partial J}{\partial \mu^t_{\text{body}}} = \frac{\partial J}{\partial \mu^t}\frac{\partial \mu^t}{\partial \mu^t_{\text{body}}} = -w_t^\top[\text{Ad}^*_{g^{t-1}}]$$

where $[\text{Ad}^*_{g^{t-1}}]$ is the 6×6 matrix representing the group co-adjoint action. One may avoid building the group co-adjoint action matrix with the following lemma.

Lemma 2. *For $g = (A, b) \in \text{SE}(3)$, the matrix-vector product of the transpose of the group co-adjoint action on a vector $\mu \in \mathbb{R}^6$ is equal to*

$$[\text{Ad}^*_g]^\top \mu = \eta\left(\text{Ad}^*_{g^{-1}}(\eta(\mu))\right),$$

where $\eta : (a, b) \mapsto (b, a)$ is an involution of first and last three components of $\mu \in \mathbb{R}^6$.

Proof. Starting from the right hand side of Lemma 2, we are interested in the group co-adjoint action with respect to the inverse rigid body transformation $g^{-1} = (A^\top, -A^\top b)$ on the involution of $\mu = (l, p) \in \mathbb{R}^6$. By definition, we have

$$\eta\left(\text{Ad}^*_{g^{-1}}(p, l)\right) = \begin{pmatrix} A^\top l \\ A^\top p - A^\top b \times A^\top l \end{pmatrix}.$$

On the other hand, the transpose of the 6×6 matrix $[\text{Ad}^*_g]$ representing the group co-adjoint action is

$$\begin{pmatrix} A & [b]_\times A \\ 0 & A \end{pmatrix}^\top = \begin{pmatrix} A^\top & 0 \\ -A^\top[b]_\times & A^\top \end{pmatrix},$$

so that its application to μ also gives

$$[\mathrm{Ad}_g^*]^\top \begin{pmatrix} l \\ p \end{pmatrix} = \begin{pmatrix} A^\top l \\ A^\top p - A^\top b \times A^\top l \end{pmatrix},$$

which concludes the proof. □

For the momentum derivative, we have

$$D\mu^{t+1}(g^t)[g'] = \left(A'l^{t+1} + b^t \times A'p^{t+1} + b' \times A^t p^{t+1}, A'p^{t+1}\right), \tag{4.8}$$

where $g' = (A', b')$, $g^t = (A^t, b^t)$ and $\mu^{t+1} = (l^{t+1}, p^{t+1})$. To explain this formula, recall that we are interested in the Euclidean directional derivative of the function

$$\mathrm{SE}(3) \to \mathfrak{se}(3)^*, \quad g \mapsto \mathrm{Ad}_g^*(\mu),$$

so consider the Euclidean extension of the group co-adjoint action on some general $\mu = (l, p) \in \mathbb{R}^6$

$$A_\mu^* : \mathbb{R}^{12} \to \mathbb{R}^6, \quad g = (A, b) \mapsto (Al + b \times Ap, Ap).$$

Then we compute the directional derivative

$$\begin{aligned} DA_\mu^*(g)[g'] &= \frac{\mathrm{d}A_\mu^*}{\mathrm{d}t}(g + t(A', b'))\bigg|_{t=0} \\ &= \lim_{t\to 0} \frac{1}{t}\left(A_\mu^*(g + t(A', b') - A_\mu^*(g))\right) \\ &= \lim_{t\to 0} \frac{1}{t}\left(tA'l + tb \times A'p + tb' \times Ap + t^2 b' \times A'p, tA'p\right) \\ &= (A'l + b \times A'p + b' \times Ap, A'p), \end{aligned}$$

which is the required formula.

Discrete world forces Another simple case is when we have a handle over world forces f^t_{world} via the formula

$$\mu^t = \sum_{\tau=2}^{t} f^\tau_{\mathrm{world}}.$$

Then the gradient of the objective function with respect to the world forces is

$$\frac{\partial J}{\partial f^t_{\mathrm{world}}} = -\sum_{\tau=t}^{T} w_\tau^\top,$$

and due to the independence of the world forces on the rigid body transformations we have

$$D\mu^{t+1}(g^t)[g'] = 0.$$

Discrete body forces When we control the body forces f^t_{body} pushing on body's centre of mass then the momentum is given by

$$\mu^t = \sum_{\tau=2}^{t} \mathrm{Ad}^*_{g^{\tau-1}}(f^\tau_{\mathrm{body}}).$$

The corresponding gradient of the objective function is

$$\frac{\partial J}{\partial f^t_{\text{body}}} = -\sum_{\tau=t}^{T} w_\tau^\top [\text{Ad}^*_{g^{\tau-1}}],$$

and the directional derivative of momentum is similar to the case of body momentum,

$$D\mu^{t+1}(g^t)[g'] = D(\text{Ad}^*_{g^t}(f^{t+1}_{\text{body}}))(g^t)[g'],$$

which is computed as in Equation 4.8.

Discrete directional forces Sometimes forces can only be directed in a particular direction $f_{\text{fixed}} \in \mathbb{R}^6$, and we have control over magnitudes $\|f^t\|$. Then the world momentum is computed via

$$\mu^t = \sum_{\tau=2}^{t} \text{Ad}^*_{g^{\tau-1}}(\|f^\tau\| f_{\text{fixed}}).$$

The gradient of the objective function can be computed through the chain rule

$$\frac{\partial J}{\partial \|f^t\|} = \frac{\partial J}{\partial f^t_{\text{body}}} \frac{\partial f^t_{\text{body}}}{\partial \|f^t\|} = \frac{\partial J}{\partial f^t_{\text{body}}} f_{\text{fixed}},$$

given that $f^t_{\text{body}} = \|f^\tau\| f_{\text{fixed}}$. As for the the directional derivative of momentum, we we compute

$$D\mu^{t+1}(g^t)[g'] = D(\text{Ad}^*_{g^t}(\|f^{t+1}\| f^{t+1}_{\text{fixed}}))(g^t)[g'].$$

with Equation 4.8, as before.

Offset and pinned forces If the forces are offset with respect the centre of mass of the object due to a (potentially constant) map from shapes to rigid body transformation $h : \mathcal{S} \to \text{SE}(3)$, then we have

$$\mu^t = \sum_{\tau=2}^{t} \text{Ad}^*_{g^{\tau-1} h(s^{\tau-1})}(f^\tau_{\text{offset}}).$$

So the gradient of J can be computed through the chain rule

$$\frac{\partial J}{\partial f^t_{\text{offset}}} = \frac{\partial J}{\partial f^t_{\text{body}}} \frac{\partial f^t_{\text{body}}}{\partial f^t_{\text{offset}}} = \frac{\partial J}{\partial f^t_{\text{body}}} [\text{Ad}^*_{h(s^{t-1})}],$$

since $f^t_{\text{body}} = \text{Ad}^*_{h(s^{t-1})}(f^t_{\text{offset}})$, and can be efficiently computed with Lemma 2. As for the the directional derivative of momentum, we compute

$$D\mu^{t+1}(g^t)[g'] = D(\text{Ad}^*_{g^t h(s^t)}(f^{t+1}_{\text{offset}}))(g^t)[g'].$$

with Equation 4.8, as before.

Chapter 5

Experiments

We demonstrate the algorithm in scenarios of increasing complexity. The full list of experiment details, including choice of design variables, computation times, and convergence information can be viewed at Table 5.1.

Figure	Variable	Bound	T	Iters	Time	Conv.	Note
5.1c	μ_{world}	-	30	170	3m	Yes	-
5.2b	μ_{world}	-	30	133	5m	Yes	-.
5.2c	μ_{body}	-	30	1000	21m	No	$\lvert\partial J/\partial\mu_{\text{body}}\rvert \simeq 10^{-5}$
5.3a	f_{world}	[-1,1]	30	512	10m	Yes	-
5.3b	f_{world}	[-1,1]	30	1000	20m	No	$\Delta J \simeq 10^{-5}$
5.4	f_{offset}	[-0.1,0.1]	30	$\simeq$30	1m	Yes	-
5.5c	f_{pinned}	[-0.1,0.1]	30	$\simeq$8	30s	Yes	Rand. Init.

Table 5.1: Table of computation times for our Python 3.11 script running on an Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz. We say that the optimisation converges if the projected gradient norm is less than 10^{-7} or if the objective function value changes by less than 10^{-12}. All variables were initialised to zero except for the experiment in Figure 5.5.

5.1 Controlling a rigid body

We begin by showing that we have control over the motion of a completely rigid body, so we focus on a shape space with a single element: a tetrahedron with uniform mass density. In Figure 5.1 the tetrahedron must reach three checkpoint transformations by controlling its world-frame momentum.

The optimal motion trajectory Figure 5.1c deviates from a perfectly straight line for two reasons. First, there is no particular constraint on the intermediate positions other than that they should eventually bring the body closer to the checkpoints. Second, for the last two checkpoints, the designed world-frame momentum must express infinitesimal translation along a line, but this is only possible if the tetrahedron additionally rotates. Therefore, designing the momentum in body-frame is more natural in this scenario. Thus in Figure 5.2 we show how L2-regularisation of momentum may be leveraged to encourage straighter paths.

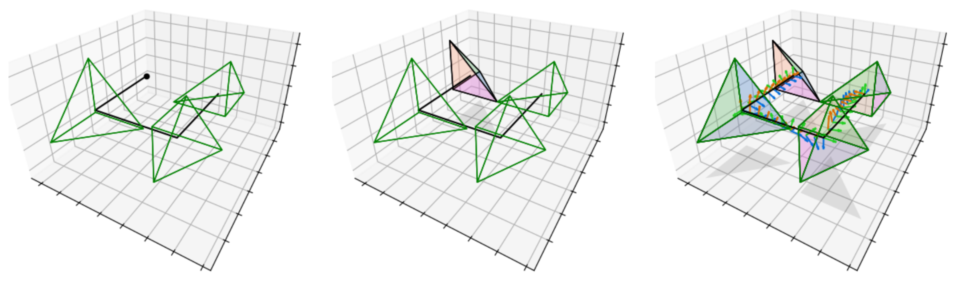

(a) Objective positions (b) Initial sequence (c) Optimised sequence

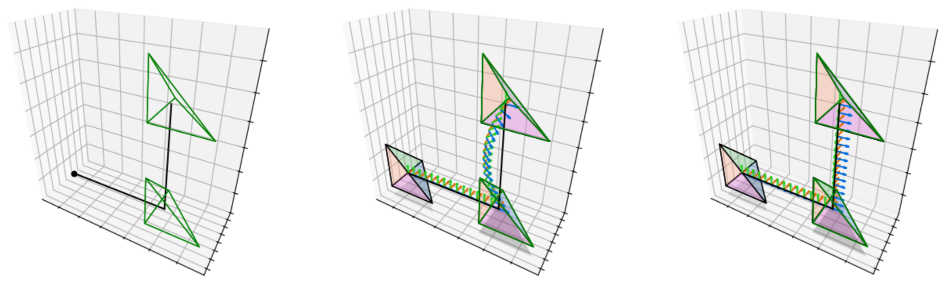
Figure 5.1: **Demonstrating the design of world-frame momentum.** The left figure (a) shows the objective, which is to reach three designated positions in intervals of 10 time steps, matching the orientations at each position. The middle figure (b) shows the initial state of the algorithm, where the momentum of is zero and thus the rigid body stays still. The right figure (c) shows the final state of the algorithm with the corresponding motion trajectory, matching the objective positions perfectly.

In physical motion design, rather than controlling momentum directly, we typically only have leverage over forces. The difference between the design of momentum and the design of forces is that a rigid body cannot instantly change its direction. Rather, a force must counteract the sum of all cumulated forces up until that point. Simultaneously, we may want to use the least amount of force, in bursts, while still achieving the motion objective. We demonstrate this for the tetrahedron in Figure 5.3 using the L1-regularisation of world forces, encouraging their sparsity.

Now, we may control body forces just as well as body momentum. In addition, we show how the optimisation can handle *offset* forces. That is, forces that are some rigid body transformation away from the centre of mass of the tetrahedron. For this, we set up an experiment in Figure 5.4 where the tetrahedron must simply move one unit forward, under different offsets.

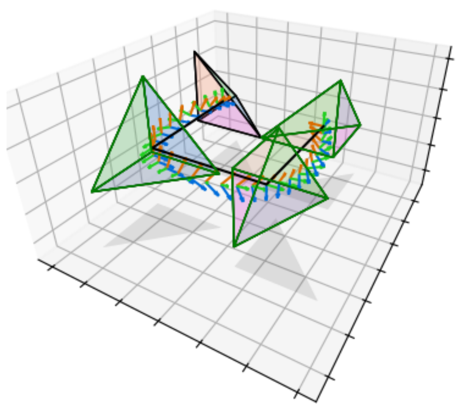
5.2 Controlling a shape-changing body

These tetrahedra have the privilege of omni-axis thrust and torque, but real-world propulsion systems are often-times merely unidirectional. This immensely constrains the set of achievable motion trajectories, unless there is some mechanism to direct the forces through shape change. In one final experiment, we describe a system that must precisely combine shape change and forces in order to follow a motion trajectory that would not be achievable by shape change or forces alone.

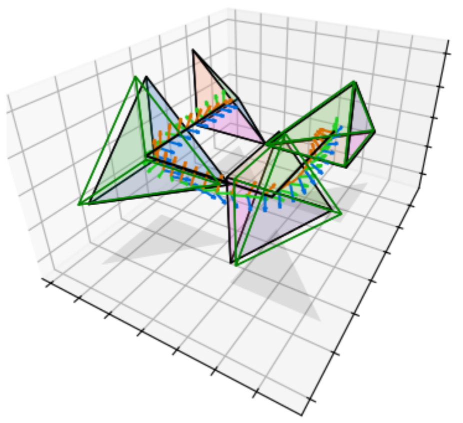


(a) Objective positions (b) Optimised μ_{world} (c) Optimised μ_{body}

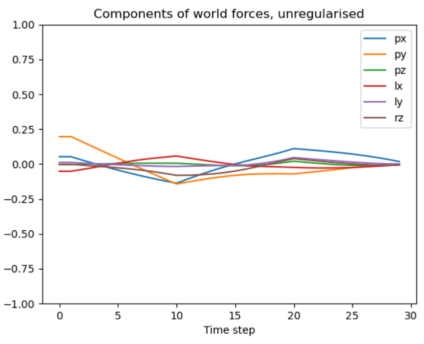
Figure 5.2: **Straightening motion paths with the design of body-frame momentum and regularisation.** The left figure (a) shows the objective, which is to reach two designated positions in an elbow arrangement in intervals of 15 time steps, matching the orientations at each position, while simultaneously minimising the L2-norm of the design variable with regularisation weight 10^{-5}. The middle figure (b) shows the converged path for world-frame momentum, where it is impossible to regularise away the rotational component of the world momentum. The right figure (c) shows the converged path for body-frame momentum, where the body-frame momentum is almost constant between checkpoints.



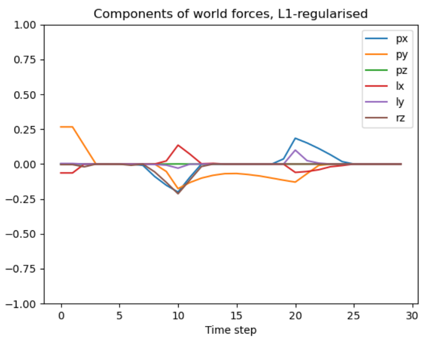
(a) Optimised f_{world} trajectory



(b) L1-regularised f_{world} trajectory

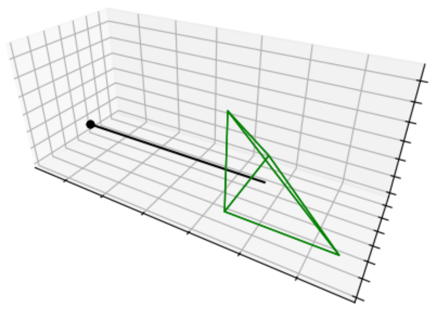


(c) Optimised f_{world} components

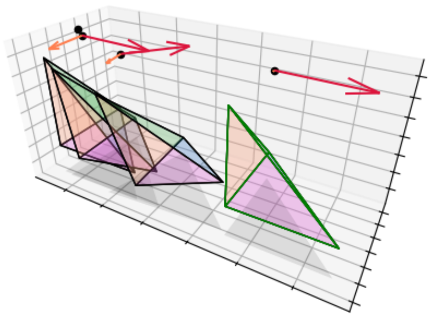


(d) L1-regularised f_{world} components

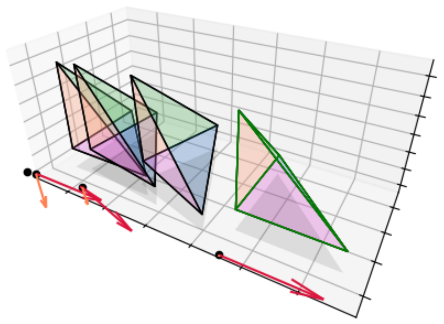
Figure 5.3: **Demonstrating the design of world-frame forces.** We use the same checkpoints objective function as in Figure 5.1a. The top row of figures shows the designed motion trajectory, and the bottom row of figures shows the components of the world-frame force vector over time. In the right hand column, of figures, an additional L1 regularisation term is added to the objective function with weight 10^{-2}. This causes a sparsity pattern to appear in figure (d), and the trajectory to tighten up overall. In the regularised case, the optimisation slowed immensely and the maximum number of iterations was reached before convergence. We only expect the sparsity pattern to get more pronounced as the computational budget increases.



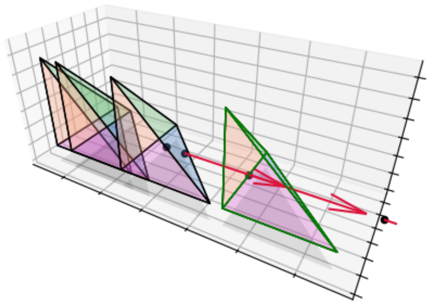
(a) Position objective



(b) Optimised f_{offset} (offset by local Z)

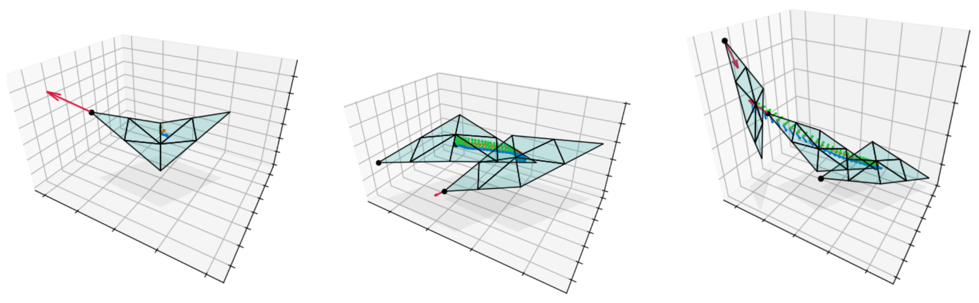


(c) Optimised f_{offset} (offset by local Y)



(d) Optimised f_{offset} (offset by local X)

Figure 5.4: **Designing offset body forces.** In the top-left figure (a) a simple position objective is described. In figures (b), (c), and (d), four frames at regular time intervals are plotted. Alongside the tetrahedron is a black anchor point that is at an offset from the centre of mass of the tetrahedron. Forces f_{offset} push against the anchor point, which in turn pushes the tetrahedron. The red arrows represent linear forces, while the orange arrows represent torque (following the right-hand rule). In the order of (b), (c), (d), the offsets are in the Z, Y, and X directions of the local body frame. The algorithm is able to balance torque and forces in order to steer the tetrahedron perfectly towards the target position.



(a) Pinned force setup (b) Initial f_{pinned} (c) Optimised f_{pinned}

Figure 5.5: **Co-design of shapes and pinned forces.** In this experiment, a flap must rotate 90 degrees about the Y-axis. In order for this to happen, it must time its forces and shape changes precisely. On one hand, if the flap is completely open, then the force cannot by itself lift the shape out of plane. On the other hand, because the flap has a single degree of freedom, it cannot generate a geometric phase to rotate itself about the Y-axis. Therefore both shape change and forces must at once for the flap to move out of plane. On the left figure (a) is the setup, with a flap shape generated by its dihedral angle and a unidirectional force pinned to the tip of one of its wings. In the middle figure (b) a random initial sequence of dihedral angles and forces are generated. In the third figure (c) the flap reaches a good 90 degree tilt.

Chapter 6

Discussion

Our results demonstrate that meaningful control can be achieved even in constrained systems limited directions of propulsion thanks to the interaction of shape change and force direction.

6.1 Limitations

Given that we build off the work of [2], we inherit many of the same limitations. Namely, there is an inherent trade-off between the motion objectives and regularisation, which can result in bad motion trajectories. Furthermore, the optimisation very often gets stuck and is unable to continue unless particular conditions are met. For example, far away position objectives tend to be more difficult to satisfy, and if the design variables are initialised randomly, the algorithm may need to be re-run multiple times before L-BFGS gets a handle on the curvature of the optimisation landscape, especially in Figure 5.5.

Future directions In our toy example of a hinged flap, we prevent self-intersection with hard angle constraints. However, animals that move by combining shape change and propulsion, like a squid, have no such hard constraints. Thus correct simulation of such an animal might incorporate a self-repelling energy term to discourage self-intersection, such as the *repulsive energy* in Sassen et al.. Otherwise, if it is interesting to move towards physical interpolation of animation frames that includes contact with rigid objects (for instance, a walk cycle), then it could be of interest to look towards the work of Li et al.. Finally, we have mostly looked at the shape-change and propulsion of singular mechanisms, but it could be interesting to discover modes of locomotion that involve multiple bodies at once.

6.2 Conclusion

We have tackled the *inverse geometric propulsion* problem. That is, we have developed and validated an efficient gradient-based algorithm for designing shape changes and momentum in order to achieve particular motion objectives. Along the way, we have drawn from ideas in differential geometry so that our method generalises to any configuration space of shapes in correspondence. Our examples validate the basic premise of the method, and serve as building blocks for potential applications in robotics and computer animation.

Bibliography

[1] ClearSpace — clearspace.today. `https://clearspace.today/`. [Accessed 25-06-2025].

[2] Becker, Q., O. Gross, and M. Pauly (2025). Inverse geometric locomotion. *ACM Trans. Graph. 44*(4).

[3] Borgefors, G. (1984). Distance transformations in arbitrary dimensions. *Computer vision, graphics, and image processing 27*(3), 321–345.

[4] Byrd, R. H., P. Lu, J. Nocedal, and C. Zhu (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing 16*(5), 1190–1208.

[5] Evans, B. (2012). “it'll be a miracle”: The rescue of palapa and westar (part 1). `https://www.americaspace.com/2012/11/17/itll-be-a-miracle-the-rescue-of-palapa-and-westar-part-1/`. Accessed June 13, 2025.

[6] Kühnel, W. (1999). *Differentialgeometrie*, Volume 2003, pp. 155–156. Springer.

[7] Lenda, J. (1978, May). Manned maneuvering unit user's guide. `https://ntrs.nasa.gov/api/citations/19790008382/downloads/19790008382.pdf`. Accessed June 11, 2025.

[8] Li, M., Z. Ferguson, T. Schneider, T. Langlois, D. Zorin, D. Panozzo, C. Jiang, and D. M. Kaufman (2020). Incremental potential contact: Intersection- and inversion-free large deformation dynamics. *ACM Trans. Graph. (SIGGRAPH) 39*(4).

[9] Marsden, J. E. and T. S. Ratiu (1999). *Introduction to Mechanics and Symmetry: A Basic Exposition of Classical Mechanical Systems.* Springer New York.

[10] Marsden, J. E. and M. West (2001). Discrete mechanics and variational integrators. *Acta numerica 10*, 357–514.

[11] Moré, J. J., B. S. Garbow, and K. E. Hillstrom (1980). User guide for minpack-1. Technical report, CM-P00068642.

[12] Sassen, J., H. Schumacher, M. Rumpf, and K. Crane (2024). Repulsive shells. *ACM Transactions on Graphics 43*(4), 1–22.