AAS/AIAA Space Flight Mechanics Meeting, Napa, CA in February 14-18, 2016

Reinforcement Learning for Spacecraft Maneuvering Near Small Bodies

Stefan Willis,* Dario Izzo,† Daniel Hennes‡

ESA's Advanced Concepts Team, Noordwijk, 2201 AZ, The Netherlands

We use neural reinforcement learning to control a spacecraft around a small celestial body whose gravity field is unknown. The small body is assumed to be a triaxial ellipsoid and its density and dimensions are left unknown within large bounds. We experiment with different proprioceptive capabilities of the spacecraft emphasising lightweight neuromorphic systems for optic flow detection. We find that even in such a highly uncertain environment and using limited perception capabilities, our approach is able to deliver a control strategy able to hover above the asteroid surface with small residual drift.

I. Introduction

The gravity field surrounding small bodies in our solar system is rather weak. The gravitational acceleration in low Earth orbit is around 9 ms^{-2}, in contrast, a small body like Rosetta's comet 67P/Churyumov-Gerasimenko produces a gravitational attraction which is six orders of magnitude smaller. Other effects often considered perturbative, like solar radiation pressure, thus become much more significant in the surrounding of these small bodies. Additionally, the highly irregular shapes of small bodies result in gravity fields which can hardly be related to the simple spherical case. Thus, from the orbital mechanics point of view, small bodies are among the most interesting and extreme environments currently found in our solar system. Investigations on the stability limits of orbits in the vicinity of small bodies[1,2] show how uncontrolled trajectories can be "unstable" and the spacecraft easily impacts or escapes the small body also within short time spans. A related but different problem to that of orbiting small bodies is that of hovering,[3] which is an option in weak gravitational fields of slowly rotating bodies as current low thrust propulsion systems allow there for quite some maneuverability. In a hovering pahse the spacecraft either keeps its position fixed with respect to the small body's surface (body-fixed hovering) or keeps it fixed with respect to an inertial fixed reference frame (inertial-frame hovering). A nearly continuous thrusting effort is needed to enact such a strategy. Hovering, as a strategy to gain control over a spacecraft's trajectory, has several applications. Body-fixed hovering can be the starting point for a landing sequence, or can allow the acquisition of high resolution imagery and measurements of a particular area on the body's surface. Applications for inertial-frame hovering, in which the small body rotates below the spacecraft, include the mapping of a small body's surface, the use as a holding orbit from which to stage further maneuver or some asteroid deflection concepts. Applying traditional control methods for close proximity maneuvering near small celestial bodies requires accurate knowledge of the small body's gravitational environment, shape and rotational state. To obtain this knowledge, the spacecraft has to fly multiple controlled trajectories around the celestial body. These controlled trajectories are costly in both, time and use of propellant.[4,5] A study by Broschart[5] on the Hayabusa mission's target asteroid Itokawa investigated the feasibility of body-fixed and inertial-frame hovering, when the dynamics of the small body are known. The study concludes that, given the mission parameters, both body-fixed and inertial-frame hovering is indeed possible. However, a controller able to achieve hovering in a small body's vicinity without knowing the environmental dynamics has the potential to save considerable time and resources. From the perspective of machine learning, the task of learning to hover in a stochastic environment can be cast as a reinforcement learning (RL) problem. Reinforcement learning is deeply rooted in psychological and neuroscientific perspectives on animal behavior. Software agents are exposed to an environment and try to learn good behavior, i.e., to take actions which optimize some notion of cumulative reward. Initially, the agent may not know anything about its best behavior but iteratively interacts with the environment and observes feedback, i.e., rewards from it. In order to

*Intern, Advanced Concepts Team, ESA

†Scientific Coordinator, Advanced Concepts Team, ESA

‡Research Fellow, Advanced Concepts Team, ESA.

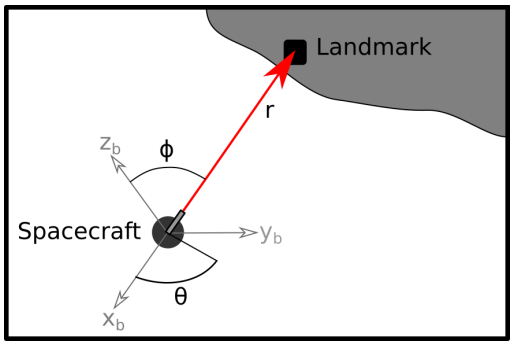

Figure 1: Acquisition of relative position offset to landmark on small body surface.

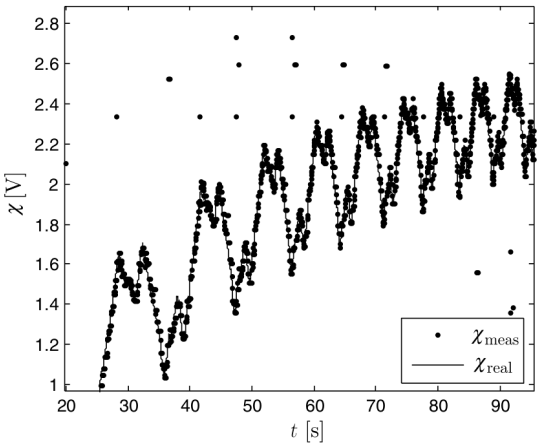

Figure 2: Simulated EMD response from an asteroid scenario, while the spacecraft is performing a spiral descent. Image taken from Izzo et al.[8]

maximize the cumulative reward, the agent adapts its behavior accordingly. According to Sutton et al.,[6] the principles of interaction, observation and optimization of a cumulative reward are the most important distinguishing features of RL.

In this work, we take the RL approach to design a controller around an unknown small body and we show how the hovering task can be solved efficiently under rather large uncertainties. We start from and extend the results reported in the work by Gaudet et al.[7] achieving higher precision in the positional control, considering a more generic gravity model (tri-axial ellipsoid) and, in addition, studying the possibility to use only neuromorphic sensors measuring optic flow observables to hover over an unknown small-body.

II. Sensory Systems

The next two sections describe the origin of the sensory inputs (i.e., the MDP state spaces) used in our experimental setup. First, the relative localization sensor system is explained. Second, the ventral optic flow and time-to-contact sensor system is explained.

A. Relative Offset and Velocity to Landmark

A possible sensory system which is able to report relative position offsets to a target landmark on the small body's surface is based on a method borrowed from proportional navigation guidance used for guided missile homing.[9] The system consists of an optical seeker which tracks a landmark on the small body's surface, and a laser range finder which is pointed in the same direction as the optical seeker. A gyroscope on board the spacecraft is used to establish a body fixed reference frame $\mathcal{F}_b = \left[\hat{\mathbf{b}}_x, \hat{\mathbf{b}}_y, \hat{\mathbf{b}}_z\right]$. This system is able to express the position of a tracked landmark in $\mathcal{F}_b$. It measures the angles θ and ϕ of the seeker's orientation with respect to the spacecraft's attitude, and obtains the distance r to the landmark from the laser range finder. Thus, the landmark's position in $\mathcal{F}_b$ can be described as

$$\mathbf{p} = \mathcal{F}_b \cdot \begin{pmatrix} r\cos\theta\sin\phi \\ r\sin\theta\sin\phi \\ r\cos\phi \end{pmatrix}.$$

If the system stores the first obtained position right after selecting the landmark, it is able to output relative position offsets further on. Instead of expressing the landmark's position in a spacecraft centered reference frame, one can also express the spacecraft's position in a landmark centered reference frame. The later is more intuitive. Once the spacecraft's relative position is obtained, its velocity can be estimated using e.g., a Kalman filter.[10] Note that in the previous work by Gaudet et al.[7] this type of relative positioning system is assumed.

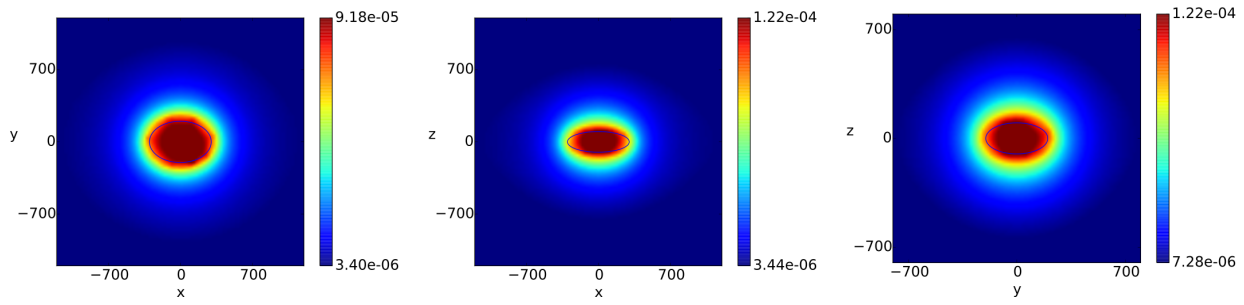

Figure 3: Gravity potential of a tri-axial body with $a = 300$, $b = 200$ and $c = 100$ [m] and a density $\rho = 2.8$ [g/cm^3] in separated 2D planes.

B. Elementary Motion Detectors

The benefits of neuromorphic electronics which mimic neurobiological circuits with the aim of replicating the circuits' functionality, are well known.[11] These electronics are realized on very-large-scale integrated (VLSI) systems using either purely analog, digital, or a mixture of both circuits. Their major features include fault- and noise-tolerance, energy efficiency, and fast processing of data with minimal mass and volume requirements. As a consequence, neuromorphic electronics find applications where these features are key requirements. For example a prosthesis which mimics the cochlea in the human ear. Another example of a neurobiological circuit that has been mimicked is the visual autopilot of insects. Research dates back into the 1960s focusing on the development of similar control systems based on elementary motion detectors (EMDs). EMDs are neuromorphic electronics which are able to detect motion from visual contrast and are implemented for more then ten years, e.g., on VLSI chips.[12] They have been used for example to implement visual autopilots for unmanned aerial vehicles.[13–15]

Another application of EMD based navigation is planetary exploration[16] and landing for which the first studies[17,18] were coordinated by the Advanced Concepts Team of the European Space Agency. The usefulness of EMD systems in a closed loop control architecture to perform planetary landing was investigated. These studies make use of a software called PANGU[19] with which realistic surface images of landing scenarios can be produced. These images were fed into the EMD sensor system to extract motion information in terms of ventral optic flow ϖ and time-to-contact τ. Geometrically, these measures can be defined as

$$\varpi = \frac{v_{\parallel}}{h} \qquad \tau = -\frac{h}{v_{\perp}}$$

where $v_{\parallel}$ is the spacecraft's velocity parallel to the surface and $v_{\perp}$ is its velocity perpendicular to the surface. The first proposal to introduce the usage of ventral optic flow and time-to-contact to planetary landing is rather recent.[20] Figure 2 shows the simulated output χ of an analog EMD circuit which is mounted on a spacecraft above an asteroid and pointed at nadir. The output is in volts and its magnitude is related to the ventral optic flow by a known curve characteristic of the EMD circuit itself. The time series of the EMD system output can be explained by the descending motion of the spacecraft and the asteroid's constant rotation speed. The closer the spacecraft is at the ground, the higher the EMD sensor response is. Thus, descending towards the surface causes the EMD sensor to measure increasingly higher ventral optic flow, which explains the increasing tendency of the EMD system output. The asteroid's irregular shape rotates uniformly below the spacecraft, which causes the height to vary periodically. This is visible in the EMD system output as a periodic volt pattern.

III. Dynamics

The small body is modeled as a tumbling tri-axial ellipsoid with uniform density ρ. The ellipsoid axes have length a, b and c. Its principal moments of inertia are defined, without loss of generality, as $I_x < I_y < I_z$. The reference frame $\mathcal{F}_S$, i.e. the small body's body axes, has its x, y, and z axes aligned to the principal axes a, b and c and its origin at the small body's center of mass. Since the small body is assumed to be tumbling with an angular velocity $\boldsymbol{\omega}(t)$, $\mathcal{F}_S$ is not an inertial reference frame.

Under these assumptions, the equations of motion of a spacecraft subject to the small body gravitational pull and

to solar radiation pressure are written, in the $\mathcal{F}_S$ frame, as:

$$\begin{aligned} \ddot{\mathbf{r}} &= -2\boldsymbol{\omega} \times \dot{\mathbf{r}} - \dot{\boldsymbol{\omega}} \times \mathbf{r} - \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}) + \mathbf{a}_g + \mathbf{a}_{SRP} + \frac{\mathbf{T}}{m} \\ \dot{m} &= -\frac{T}{I_{sp} g_0} \end{aligned} \tag{1}$$

where $2\boldsymbol{\omega} \times \dot{\mathbf{r}}$ is the Coriolis acceleration, $\dot{\boldsymbol{\omega}} \times \mathbf{r}$ is the Euler acceleration, $\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r})$ is the centrifugal acceleration, $\mathbf{a}_g$ is the gravitational acceleration due to the tri-axial ellipsoid, $\mathbf{a}_{SRP}$ is the acceleration caused by solar radiation pressure and unknown random perturbations, $\mathbf{T}$ is the commanded thrust, I_{sp} the propulsion system specific impulse and $g_0 = 9.80655$ [m/s^2]. Note that in the above equations, the angular velocity $\boldsymbol{\omega}(t)$ is, in general, an explicit function of time. Fortunately, under the assumption of rigid body motion, such a function can be elegantly expressed in terms of the Jacobian elliptic functions cn, sn and dn:[21]

$$\begin{aligned} \omega_x(t) &= \sqrt{\frac{2EI_z - M^2}{I_x(I_z - I_x)}}\mathrm{cn}\tau \\ \omega_y(t) &= \sqrt{\frac{2EI_z - M^2}{I_y(I_z - I_y)}}\mathrm{sn}\tau \\ \omega_z(t) &= \sqrt{\frac{M^2 - 2EI_x}{I_z(I_z - I_x)}}\mathrm{dn}\tau \end{aligned}$$

where,

$$\tau = t\sqrt{\frac{(I_z - I_y)(M^2 - 2EI_x)}{I_x I_y I_z}}$$

and E is the energy of the system, which is conserved, and has the expressions,

$$I_x\omega_x^2 + I_y\omega_y^2 + I_z\omega_z^2 = 2E$$

M is the magnitude of the system angular momentum which is also conserved and canis expressed by,

$$I_x^2\omega_x^2 + I_y^2\omega_y^2 + I_z^2\omega_z^2 = M^2$$

The elliptic modulus of the Jacobian elliptic functions is,

$$k^2 = \frac{(I_y - I_x)\left(2EI_z - M^2\right)}{(I_z - I_y)(M^2 - 2EI_x)}.$$

In all the above formulas, for simplicity, $\omega_y(0) = 0$ is assumed so that only two remaining initial conditions can be specified. It is thus possible also to only specify E and M, which, in turn, define the starting small body's angular velocity as $\boldsymbol{\omega}_0 = [\omega_x(0), 0, \omega_z(0)]$. Note that, while the small body will not necessarily have a periodic motion, the angular velocity has and its period P is given by,

$$P = 4K\sqrt{\frac{I_x I_y I_z}{(I_z - I_y)(M^2 - 2EI_x)}} \tag{2}$$

where K is a complete elliptic integral of the first kind. After computing $\boldsymbol{\omega}(t)$ at time t, the angular acceleration $\dot{\boldsymbol{\omega}}(t)$ at time t can be computed by substituting $\boldsymbol{\omega}(t)$ into Euler's equations

$$\begin{aligned} \dot{\omega}_x &= -\frac{(I_z - I_y)\omega_y\omega_z}{I_x} \\ \dot{\omega}_y &= -\frac{(I_x - I_z)\omega_z\omega_x}{I_y} \\ \dot{\omega}_z &= -\frac{(I_y - I_x)\omega_x\omega_y}{I_z} \end{aligned}$$

The analytic solution allows computing the angular components in constant time without loss of precision. The alternative is to integrate Euler's rotation equations, which makes the precision and evaluation time depend on the time that passed since the start of the simulation.

1. Gravity

The term $\mathbf{a}_g$ in Eq.(1) is the gravitational acceleration coming from the tri-axial ellipsoid. Since the small body's density ρ is uniform, the gravitational parameter μ for the small body can be expressed as,

$$\mu = G\rho\frac{4}{3}\pi abc$$

where $G = 6.6695 \times 10^{-11}$ [m^3 / kg / s^2] is the universal gravitational constant. The gravitational potential V of a solid homogeneous tri-axial ellipsoid in $\mathcal{R}_S$ can be expressed as:

$$V(x,y,z) = \frac{3}{4}\mu\int_{\kappa_0}^{\infty}\left(1 - \frac{x^2}{a^2+\kappa} - \frac{y^2}{b^2+\kappa} - \frac{z^2}{c^2+\kappa}\right)\cdot \cdot \frac{d\kappa}{\sqrt{(a^2+\kappa)(b^2+\kappa)(c^2+\kappa)}} \tag{3}$$

where κ_0 is the largest root of the confocal ellipsoid C defined as

$$C(\kappa) = \frac{x^2}{a^2+\kappa} + \frac{y^2}{b^2+\kappa} + \frac{z^2}{c^2+\kappa} - 1.$$

The above expressions are attributed to Ivory,[22] a detailed derivation is given by Mac Millan,[23] Danby[24] and Moulton.[25] The integrals appearing in Eq.(3), are elliptic integrals. Introducing Legendre's canonical elliptic integrals of first kind F and second kind E it is possible to find an explicit expression for V and its derivatives (e.g. the gravitational acceleration). The complete derivation of these expressions can be found in the work of Cersosismo.[26] Here, though, we develop and use an alternative approach which relates the gravitational potential to the more modern Carlson elliptic integrals.
Introduce the following variable transformation $\kappa' = \kappa - \kappa_0$ in Eq.(3). The expression for the gravitational potential of a tri-axial ellipsoid becomes,

$$\begin{aligned} V(x,y,z) = &\frac{3}{2}\mu R_F\left(a^2+\kappa_0, b^2+\kappa_0, c^2+\kappa_0\right) \\ &- \frac{1}{2}\mu x^2 R_D\left(b^2+\kappa_0, c^2+\kappa_0, a^2+\kappa_0\right) \\ &- \frac{1}{2}\mu y^2 R_D\left(a^2+\kappa_0, c^2+\kappa_0, b^2+\kappa_0\right) \\ &- \frac{1}{2}\mu z^2 R_D\left(a^2+\kappa_0, b^2+\kappa_0, c^2+\kappa_0\right). \end{aligned}$$

where $R_F(x,y,z)$ and $R_D(x,y,z)$ are two of the Carlson symmetric forms The explicit cartesian components of the gravitational acceleration $\mathbf{a}_g$ in (x,y,z) can than be derived by taking the gradient of V which yields,

$$\begin{aligned} a_x = \tfrac{\partial V}{\partial x} &= -\mu x R_D\left(b^2+\kappa_0, c^2+\kappa_0, a^2+\kappa_0\right) \\ a_y = \tfrac{\partial V}{\partial y} &= -\mu y R_D\left(a^2+\kappa_0, c^2+\kappa_0, b^2+\kappa_0\right) \\ a_z = \tfrac{\partial V}{\partial z} &= -\mu z R_D\left(a^2+\kappa_0, b^2+\kappa_0, c^2+\kappa_0\right). \end{aligned} \tag{4}$$

Note that when computing this gradient, one must consider k_0 as a function of x, y, z, thus deriving Eq.(4) is not as trivial as it seems. The computations can be simplified starting from Eq.(3), performing the derivatives and only then introducing Carlson symmetric forms.

The newly derived expressions express the gravity in terms of Carlson symmetric forms of elliptic integrals rather than in terms of Legendre forms of incomplete elliptic integrals. A comparison based on the GSL implementations of elliptic integrals shows that the new expressions have a substantially comparable computing complexity (speedup of x1.01609), while being substantially shorter and more elegant.

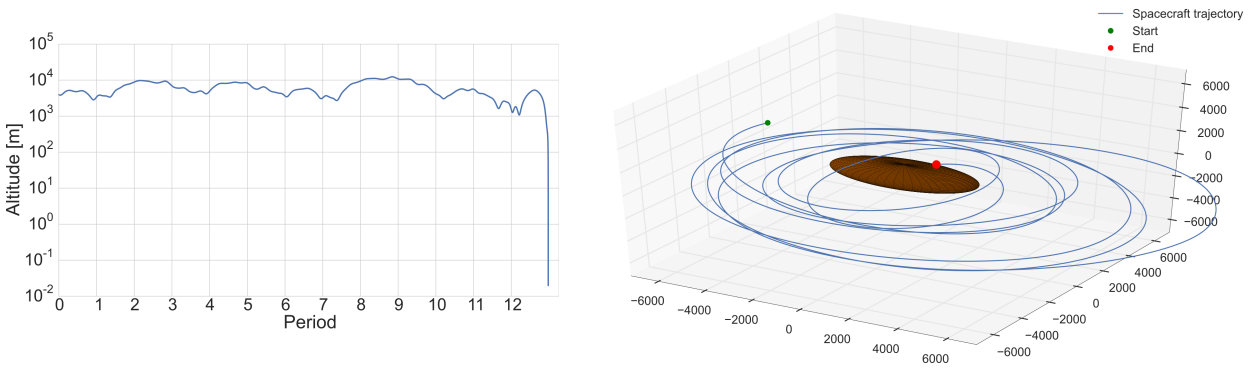

Figure 4: Example of a simulated trajectory of an uncontrolled spacecraft around a tumbling tri-axial small body. The altitude over the surface is shown (left) against the simulation time normalized by the angular velocity period P together with the actual trajectory (right).

2. *Perturbations*

The $\mathbf{a}_{SRP}$ term in Eq.(1) represents the acceleration caused by solar radiation pressure which we here account for by adding an unknown random perturbation. The acceleration is assumed to be normally distributed with nonzero mean μ_{SRP} and standard deviation σ_{SRP}

$$\mathbf{a}_{SRP} \sim \mathcal{N}\left(\mu_{SRP}, \sigma_{SRP}\right)$$

Uncertainties are also considered on the propulsion system specific impulse by setting,

$$I_{sp} = I_{sp_0} + I_{sp_0} \cdot \mathcal{N}\left(0, \sigma_{I_{sp}}\right)$$

3. *Simulation*

When simulating the equations of motion Eqq.(1), we sample the uncertain variables at 1 Hz and thus transform the equations into an ordinary differential system of equations. An adaptive time step Cash-Karp Runge Kutta integrator is then used to produce the new spacecraft state after each second. The spacecraft altitude over the small body surface is also computed along the simulation as that is ultimately the quantity we wish to control. To illustrate the complexity of the dynamics thus generated we show in Figure 4 an example of simulated trajectory in the case of an uncontrolled spacecraft. The altitude is also reported. The simulation is stopped when the spacecraft impacts the small-body surface.

IV. Evolutionary Direct Policy Search

1. *Markov Decision Process*

The controller optimization can be formulated as a Markov Decision Process (MDP) learning problem. The sensory output $\mathbf{R}$ for the assumed sensor system generates the state space $\mathcal{S}$ in the MDP. Its action space $\mathcal{A}$ is defined as $\mathbf{T}$, the spacecraft's thrust in x, y, and z dimension. Note that the thrust vector is defined in the body fixed frame and thus its actual implementation would require an attitude controller or thrust vectoring system. The transition model is defined by the equations of motion from Eqq.(1).
The rewards for state-action pairs and thus the utility functions depend on the sensor system setups. The utility function $U\left(\pi\right)$ is based on the output of the simulator $\text{sim}\left(\chi, \theta\right)$. The MDP can be constructed by a single seed χ which defines the initial conditions of the spacecraft and all properties of the small body. All actual definitions are given in the next chapter where we discuss the experimental setups.

2. *Policy Architecture*

We perform direct policy search (DPS) with a genetic algorithm to obtain controller policies with high utility. To perform DPS we have to decide on a policy architecture for which the genetic algorithm will optimize its parameters.

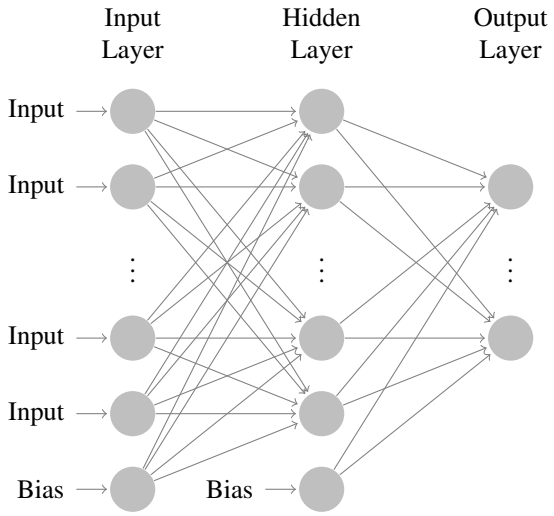

Figure 5: General architecture of a feed forward neural network with one hidden layer.

The universal approximation property of Feed Forward Neural Networks (FFNNs), their simple architecture and their short evaluation time make them a promising choice. The architecture is also able to handle continuous state and action spaces. The input dimension of the FFNN is then given by the MDP state dimenions plus a bias neuron, the output dimension of the FFNN is given by the action space, i.e., the thrust vector.

What is left to decide on is the number of hidden layers, the activation functions in the hidden and output layers, and the number of hidden neurons in each hidden layer. These are the hyperparameters of the model and their choice is nontrivial. Adding a hidden layer of dimension k after a layer of dimension l adds $k \times l + k$ parameters to the model. Therefore, the aim is to keep the number of hidden layers and hidden nodes small. Several works[7,27,28] experimenting with neural networks as policy architecture, use one hidden layer. Since we aim to compare our controllers to the one from,[7] we use one hidden layer as well. Furthermore, the policy from Gaudet et al.[7] uses two hidden neurons for each of the three independent controllers. We therefore use six hidden neurons in the hidden layer. Layers are fully connected and no human pre or post optimization of the architecture is performed.
The sigmoid function is used in the hidden and output layer, where the sigmoid function in the output layer serves as a thrust limiter to keep the FFNN's output in $(0, 1)$ which can then be scaled by the maximum magnitude of the spacecraft's thrusters.

3. Generational Particle Swarm Optimizer

For this architecture, the policy's parameters θ are the weights between the neurons in the FFNN. We encode the policy parameters in a genom by reshaping the weight matrices in the FFNN into a vector of weights and use it in our genetic algorithm. We chose a generational particle swarm optimizer as the optimization algorithm. The algorithm is based on the original PSO.[29] It is an evolutionary optimization algorithm for which promising results in evolutionary policy optimization exist.[28,30–32]

The optimization problem's cost function which defines an individual's fitness has yet to be defined. The aim is to evolve policies which have a high utility. Therefore, the fitness of an individual is defined as the utility of the policy that the individual's genom encodes. The utility function $U_{\chi_i}(\pi_\theta)$ uses the output of the simulation $\text{sim}(\chi_i, \theta)$ to compute the utility of policy π_θ. The output of this function for a given seed χ is deterministic and depends only on the policy's behaviour defined by θ.
The spacecraft's initial conditions and asteroid properties change after every generation according to the distribution of the environment parameters as described earlier. Thus, the policy evolution is learning conditioned on these distributions. Algorithm 1 shows the pseudocode of the PSO DPS algorithm. After N generations, the individual with the best fitness is picked from the population. It represents the best policy $\hat{\pi}$ which was found during evolution.

Algorithm 1 Direct policy search with a generational particle swarm optimizer.

Initialize a population of particles $P = \{p_0, \cdots, p_n\}$ with random speed and random positions in the search space $\mathcal{D}$
Initialize a random number generator τ
Initialize the champion $c \leftarrow p_0$
Sample the first k random seeds $\Xi = \{\chi_1, \cdots, \chi_k\}$ from τ
for Each particle p_i **do**
    p_i encodes a policy π_i
    Evaluate the fitness $f(\Xi, p_i) = -\frac{1}{k}\sum_{j=1}^{k} U_{\chi_j}(\pi_i)$
    if $f(\Xi, p_i) < f(\Xi, c)$ **then**
        $c \leftarrow p_i$
for N generations **do**
    Sample the next k random seeds $\Xi = \{\chi_1, \cdots, \chi_k\}$ from τ
    for Each particle p_i **do**
        Adapt velocity of the particle
        Update the position of the particle
        p_i' is the updated particle
        if $f(\Xi, p_i') < f(\Xi, p_i)$ **then**
            $p_i \leftarrow p_i'$
        if $f(\Xi, p_i') < f(\Xi, c)$ **then**
            $c \leftarrow p_i'$
Particle c encodes the best policy after N generations

V. Experimental Setups

1. Markov Decision Process Properties

The MDP is constructed with a single seed, based on which all parameters for the small body and the spacecraft are sampled. Table 1 gives an exhaustive overview of all configurable parameters and their distributions.
The maximum magnitude of the angular velocity ω depends on the size of the small body. Since most small bodies are piles of boulder, they would fly apart if the centrifugal force exceeded the gravitational force inside the small body. Let us assume a spherical model of the small body for a moment. An estimate of the maximum magnitude of ω can then be defined as follows

$$\omega^2 r \leq \frac{GM}{r^2} = \frac{G\frac{4}{3}\pi abc\rho}{r^2}$$
$$\omega \leq \sqrt{\frac{G\frac{4}{3}\pi abc\rho}{r^3}}.$$

By setting $r = a$, an upper bound on the angular velocity can be set as

$$\omega \leq \sqrt{\frac{G\frac{4}{3}\pi bc\rho}{a^2}}. \tag{5}$$

Solar radiation pressure and unknown random perturbations are defined by two parameters. Since the total of these two forces is assumed to be Gaussian, a mean μ_{SRP} and a standard deviation σ_{SRP} fully define it. These two parameters are the only environmental parameters which are fixed for all constructions.
The spacecraft is defined by six parameters. The first parameter defines its mass m, which implicitly also defines its empty mass $\frac{m}{2}$ (after all propellant has been used). The next three parameters describe the spacecraft's engine properties by a specific impulse I_{sp}, its noise $\sigma_{I_{sp}}$, and the maximum thrust T it can deliver. The spacecraft's sensor system is assumed to obtain imperfect information which is modeled by the sensor noise parameter σ_{SN}. Fixed spacecraft parameters for all environments are the engine and sensor system properties.

The spacecraft needs to be positioned around the small body. The positioning is performed using two parameters, s_{min} and s_{max}. First, a position uniformly at random on the ellipsoid is sampled, which is based on the same principle as

Parameter	Units	Distribution
Axis Scale c s_c		$\mathcal{U}(100, 8000)$
Axis Scale b s_b		$\mathcal{U}(1.1, 2)$
Axis Scale a s_a		$\mathcal{U}(1.1 \cdot s_b, 4)$
Semi Axis c	m	s_c
Semi Axis b	m	$s_b \cdot c$
Semi Axis a	m	$s_a \cdot c$
Density ρ	kgm^{-3}	$\mathcal{U}(1500, 3000)$
Angular Velocity in x direction ω_x	s^{-1}	$\mathcal{U}\left(\frac{\omega}{2}, \omega\right)$ (Eq.(5))
Angular Velocity in z direction ω_z	s^{-1}	$\mathcal{U}\left(\frac{\omega}{2}, \omega\right)$ (Eq.(5))
Time Bias t_{bias}	s	$\mathcal{U}(0, 12 \cdot 60 \cdot 60)$
Solar Radiation Pressure $\mathbf{F}_{\text{SRP}}$	N	$\mathcal{N}\left(10^{-4}, 10^{-5}\right)$
Hovering Position $\mathbf{p}$	m	Algorithm 2 with $s_{\text{min}} = 1.1$, $s_{\text{max}} = 4$
Spacecraft Mass m	kg	$\mathcal{U}(450, 500)$
Spacecraft Maximum Thrust T	N	21
Spacecraft Specific Impulse I_{sp}	s	200
Spacecraft Specific Impulse Variation $\sigma_{I_{sp}}$		0.05
Spacecraft Sensor Noise σ_{SN}		0.05

Table 1: Distributions of the initial conditions. A very rich set of different initial conditions can be generated.

Algorithm 2 Sampling a point outside an ellipsoid with semi-principal axes a, b and c.

function SAMPLEPOSITION(a,b,c,s_{min}, s_{max})

$u \leftarrow 2\pi \cdot \mathcal{U}(0, 1 - \epsilon)$

$v \leftarrow \cos^{-1}(2 \cdot \mathcal{U}(0, 1) - 1)$

$s \leftarrow \mathcal{U}(s_{\text{min}}, s_{\text{max}})$

$$\mathbf{p} \leftarrow \begin{pmatrix} s \cdot a \cdot \cos u \sin v \\ s \cdot b \cdot \sin u \sin v \\ s \cdot c \cdot \cos v \end{pmatrix}$$

return p

uniform point sampling on a sphere. The position is then moved away from the surface using a scaling factor s which is sampled uniformly from $[s_{\text{min}}, s_{\text{max}}]$. With this method, the hovering target position $\mathbf{p}$ is obtained. Depending on the sensory input, the spacecraft is initially either positioned directly at this position or starts with an offset. The spacecraft's initial velocity $\dot{\mathbf{r}}$ is distributed uniformly in each dimension as follows

$$\dot{\mathbf{r}}_i \sim \mathcal{U}(-0.3, 0.3) \qquad \forall i \in \{1, 2, 3\}$$

Note that a spacecraft with a velocity of $\dot{r} = 0$ is not moving with respect to the small body's surface, as the velocity is expressed in the body fixed frame $\mathcal{R}_S$. Similarly, the hovering target position $\mathbf{p} \in \mathcal{R}_S$ is not moving with respect to the small body's surface and is therefore fixed during a particular simulation instance. It's the uncontrolled spacecraft that drifts away from the hovering position due to the present accelerations in the non-inertial frame.

The integration step size for the simulator's ODE system is the last parameter which has to be chosen. A small integration step size is preferred as it implicitly defines the spacecraft's control frequency. However, small step sizes slow the simulation which reflects on the evaluation time of a policy's utility in the DPS algorithm. We made a compromise and set the control frequency to 1Hz, i.e., the integration step size is 1s.

A sampling was performed to get an estimate of the variety of environments in which the policies are learned. Table 2 shows the result of 10^6 sampled small body properties and spacecraft initial conditions. The largest semi-principal axis a ranges from a hundred meters to more than 30 kilometers. The spacecraft's starting position ranges from just a

	Small Body			
	Mean	Stdev	Min	Max
Estimated Period $\tilde{P}$ [s]	18315.64	5669.58	6198.40	49205.09
Semi-Principal Axis a [m]	11547.09	7242.81	132.63	31954.20
Semi-Principal Axis b [m]	6274.37	3737.17	111.32	15989.30
Semi-Principal Axis c [m]	4047.61	2281.71	100.00	8000.00
Density ρ [kgm^{-3}]	2250.61	433.11	1500.00	3000.00
	Spacecraft			
	Mean	Stdev	Min	Max
Spacecraft Height [m]	11069.05	10678.43	10.77	89667.90
Spacecraft Velocity [ms^{-1}]	0.29	0.08	0.00	0.52
Spacecraft Mass [kg]	475.00	14.43	450.00	500.00

Table 2: Initial conditions for 10^6 samples.

few meters above the surface to almost 90 kilometers away.

If the small body had a uniform rotational motion with magnitude ω', the period would be

$$\frac{2\pi}{\omega'}.$$

However, our small body model has a nonuniform rotational motion and therefore never returns to its original orientation with respect to a fixed inertial frame. Thus, we define a surrogate of the rotation period $\tilde{P}$ for a given initial angular velocity $\omega(0)$ as follows

$$\tilde{P} = \frac{2\pi}{\omega(0)}$$

which we use to set the simulation time in the DPS utility evaluation.

A. Sensory Input Simulation

1. Relative Offset and Velocity to Landmark

With the sensor system described in A, the offset to the initial starting position can be obtained. Given the spacecraft's position and velocity $\mathbf{r}$, $\dot{\mathbf{r}}$, and the target hovering position $\mathbf{p}$ defined during the MDP construction, the six dimensional MDP state produced by the sensor simulator is

$$s = \begin{pmatrix} \mathbf{p} - \mathbf{r} \\ \dot{\mathbf{r}} \end{pmatrix} \tag{6}$$

where each state variable s_i gets perturbed due to the sensor noise model as follows

$$s_i = s_i + s_i \cdot \mathcal{N}(0, \sigma_{SN}) \tag{7}$$

where σ_{SN} is defined in Table 1. If successful, a spacecraft which is maneuvered by a controller with this sensory input is expected to hover without drift, since the accumulating target position offset will be visible in the sensory input.

2. Ventral Optic Flow and Time-To-Contact

Section B described how one can obtain ventral optic flow and time-to-contact information with an EMD sensor system. Our system is assumed to be always pointing at nadir. With this assumption, ventral optic flow $\varpi_\perp$ and divergence $\varpi_\parallel$ can be geometrically defined using the spacecraft's velocity $\dot{\mathbf{r}}$ and height $\mathbf{h}$. Since the spacecraft is a

No.	Description	Sens. Input	Utility Function
DPS1	Previous Work Comparison	$\left(\Delta\mathbf{r}, \dot{\mathbf{r}}\right)$	$-\frac{1}{n-b}\sum_{i=b}^{n}\left(\|\Delta\mathbf{r}_i\| + \|\dot{\mathbf{r}}_i\|\right) + C$
DPS2	Velocity Utility Function	$\left(\Delta\mathbf{r}, \dot{\mathbf{r}}\right)$	$-\frac{1}{n-b}\sum_{i=b}^{n}\left(\|\dot{\mathbf{r}}_i\|\right) + C$
DPS3	Velocity Sensory Input	$\left(\dot{\mathbf{r}}\right)$	$-\frac{1}{n-b}\sum_{i=b}^{n}\left(\|\dot{\mathbf{r}}_i\|\right) + C$
DPS4	Optic Flow and Divergence	$\left(\varpi_{\parallel}, \varpi_{\perp}\right)$	$-\frac{1}{n-b}\sum_{i=b}^{n}\left(\|\dot{\mathbf{r}}_i\|\right) + C$

Table 3: Experimental DPS setups.

point mass and has no attitude, both sensor readings are expressed as vectors in $\mathcal{R}_S$. Let $\dot{\mathbf{h}}$ be the spacecraft's vertical velocity by projecting its velocity $\dot{\mathbf{r}}$ onto the height direction $\mathbf{h}$ as follows

$$\dot{\mathbf{h}} = \frac{\dot{\mathbf{r}} \cdot \mathbf{h}}{hh}\mathbf{h}.$$

Time-to-contact τ as described in Section B is the time until an object crosses an infinite plane defined by the image plane. Here we define the time-to-contact vector τ as a vector having magnitude equal to τ along the direction of $\mathbf{h}$ and being defined as

$$\tau = \frac{\mathbf{h}}{\dot{h}}.$$

This definition contains a singularity when the spacecraft is not moving along its height direction. Therefore, we use the divergence vector instead of the time-to-contact vector. It is defined as

$$\varpi_{\parallel} = \frac{\dot{\mathbf{h}}}{h}$$

where h will be guaranteed to be nonzero (zero height implies a crash into the surface). Furthermore, we define the ventral optic flow vector as follows

$$\varpi_{\perp} = \frac{\dot{\mathbf{r}} - \dot{\mathbf{h}}}{h}.$$

With these definitions of divergence and ventral optic flow, the MDP state produced by the sensor simulator is

$$s = \begin{pmatrix} \varpi_{\parallel} \\ \varpi_{\perp} \end{pmatrix} \tag{8}$$

where the same state variable perturbation as in Eq.(7) is applied due to the sensor noise model. The spacecraft is expected to drift because the sensory input contains no information about relative offset to the starting position and the controller is intuitively not able to recover from its drift.

B. Utility and Reward Functions

The goal is to start with the setup from previous work, and gradually work towards a controller based on only EMD sensory input. We set up several experiments to evaluate the changes in control performance. The first experiment with DPS is fairly similar to previous work[7] and establishes the performance of our derived methodology. The spacecraft's starting position $\mathbf{r}$ is assumed to have drifted away from the target hovering position $\mathbf{p}$. To model this drift, a uniform random offset is added to each dimension of $\mathbf{r}$ as follows

$$\mathbf{r}_i \sim \mathcal{U}(-3, 3) \qquad \forall i \in \{1, 2, 3\}.$$

The possible starting positions in this experiment are therefore distributed in a cubic shape around $\mathbf{p}$. The controller is then forced to correct the offset. The starting index b in the utility function is designed to give the controller enough time to perform this correction maneuver (also referred to as the transient response). For all experiments, b is set to 150 which equals 150 seconds since the integration step in the simulator is 1s.

For the remaining experiments, the spacecraft starts at its target hovering position, i.e., $\mathbf{r} = \mathbf{p}$. Body-fixed hovering implies the spacecraft to have zero velocity with respect to the surface. Therefore, we perform a second experiment with a utility function which rewards policies having close to zero velocities independently of their position. The sensory input, however, is still the same as in the first experiment.

In a next step, the relative offset to the target position information is removed from the sensor system. This sensory input can be obtained by the same sensor system as in the previous experiment with the difference of not feeding the controller with relative target position offset information. It might look strange to deny the controller this information but the experiment is designed to give a baseline for the next experiment which is based solely on the EMD sensor system. Drift is expected as the controller intuitively is not able to cancel the velocity perfectly. One can think of two reasons. First, the controller is reactive, thus the commands given for the current sensor input lack behind the actual velocity. Second, the low control frequency of 1Hz leads to more drift as one command is kept during one second while a better command resulting in less drift could be issued during this time. The controller is also not able to recover from drift as the sensory input at the next control step contains no information about the spacecraft's previously accumulated drift.

The next best information about the spacecraft's motion after relative position information is intuitively the spacecraft's actual velocity. Ventral optic flow and divergence contain information about the spacecraft's motion except that they depend on the height and the spacecraft is assumed to have no sensor system on board to establish its altitude. By keeping the ventral optic flow and divergence as close as possible to zero, the controller hopefully hovers similarly to the velocity based controller. Intuitively, the EMD system controller is not able to perform better than the controller which knows its relative velocity.

Table 3 gives an overview of all performed experiments where the relative position offset $\mathbf{p}-\mathbf{r}$ has been abbreviated by $\Delta\mathbf{r}$. Note how all utility functions include a term C. It is defined as

$$C = \begin{cases} 10^{30}, & \text{if the spacecraft crashes or runs out of fuel} \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

C. Particle Swarm Optimization

A population of 504 individuals are evolved on an archipelago[33] of 24 islands for $N = 1000$ generations, each generation containing $k = 10$ evaluations. This gives a total of 10000 different environments in which policies have to maximize their utility. The total simulation time for each simulation is set dynamic and is proportional to the small body's surrogate rotation time as follows

$$t_{\text{sim}} = \frac{\pi}{\omega\left(0\right)}$$

where $\omega\left(0\right)$ is the small body's initial angular velocity. Note that instead of using a simulation time proportional to this surrogate, another simulation time could have been set with respect to the small body's angular velocity periodicity which is defined in Eq. (2). Experiments showed, however, that such simulation times are much longer and have a wider range, which would have slowed down the fitness evaluations.

D. Performance Evaluation

Since the first two experiments can be compared to previous work, the same two performance measures are used. In particular, the Steady State Mean Error (SSME) and Steady Date Delta Error (SSDE) are used to describe the performance of the DSP policy for the first experiment.

For a trajectory $\tau = \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_n\}$ produced by a policy, the SSME is the spacecraft's average offset to the target hovering position after the transient response is assumed to have died out. It is defined as

$$\text{SSME}\left(\tau\right) = \frac{1}{n-b}\sum_{i=b}^{n}\left\|\mathbf{p}-\mathbf{r}_i\right\| + C$$

where $\mathbf{p}$ is the target hovering position, $b = 150$ is the transient response window, and C is defined in Eq.(9).
The SSDE of a trajectory τ is the difference between the largest and the smallest offset to the target hovering position

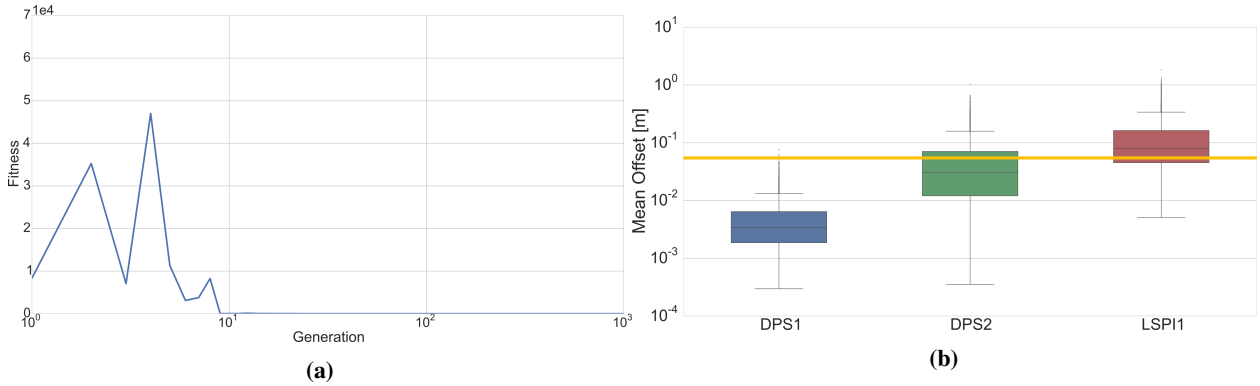

Figure 6: a) Evolution of the policy fitness for the first experiment. The population already contains a version of the best policy after less than 20 iterations.
b) Post evaluation for the task with relative position and velocity sensory input. The orange line represents the average performance of previous work.

after the transient response is assumed to have died out. It is defined as

$$\mathrm{SSDE}\left(\tau\right) = \max_{\mathbf{r}\in\tau_b} \|\mathbf{p}-\mathbf{r}\| - \min_{\mathbf{r}\in\tau_b} \|\mathbf{p}-\mathbf{r}\| + C$$

where τ_b does not contain the transient response window of size b.

The second two experiments are evaluated on a similar performance measure. Since the spacecrafts start at their target hovering position, no transient response has to be corrected. The performance measure is therefore the average offset to the spacecraft's initial position and no transient response window is given.
We postevaluated every obtained policy to get an estimate of their performance. The post evaluation process tested a given policy in 10^5 random initial conditions by simulating each MDP instance together with the policy for one hour, and then computing the performance measures based on the trajectory produced by the policy.

VI. Experimental Results

A. Controller Evaluations

Evolving a population of 504 individuals for 1000 generations with PSO takes on average 33 hours. The population is evolved on an archipelago with 24 islands, each containing 21 individuals. Each generation takes about 2 minutes, thus each MDP simulation takes on average about 0.6 seconds (one MDP simulation contains on average about 9000 seconds of simulated time). Figure 6a shows the convergence characteristics for the best policy on the first experiment. Local optima can happen when consecutive random seeds in the evolution process define similar environments. The best policy becomes overfitted to these specific environments. At some point, the active seed generates a very different environment, causing the overfitted policy to perform poorly. Thus, its fitness decreases and another individual is chosen as the new best policy.
Table 4 and Figure 6b give a performance overview for the first two experiments in terms of the performance measures used by Gaudet et al.[7] The DPS1 controller outperforms the controller developed in previous work by an order of magnitude in terms of SSME. In terms of SSDE, the controller from previous work achieves a better performance. This could be caused by the low control frequency of 1Hz used in our current setup. The DPS2 controller behaves similar to the DPS1 controller. Apparently, the controller takes advantage of the relative position information to drive the velocity as close as possible to zero. Although drift is not punished by the utility function, the controller keeps the spacecraft at its initial position with low offset.
To compare DPS with other RL algorithms, we implemented a Least Squares Policy Iteration (LSPI) RL approach[34] to learn the hovering task. The controller obtained with our LSPI methodology manages to keep the spacecraft in the vicinity of the target hovering position as well. However, the average offset to the target hovering position is two orders of magnitude higher compared to our best DPS solution.
Table 5 and Figure 7 give a performance overview for the second two experiments. Since the spacecrafts start without

	SSME [m]			
	Mean	Stdev	Min	Max
Gaudet[7]	0.06229	0.04142	0.01095	0.34487
LSPI1 Controller	0.13642	0.15275	0.00499	1.81269
DPS1 Controller	0.00519	0.00542	0.00030	0.07614
DPS2 Controller	0.05686	0.07348	0.00035	1.01809
	SSDE [m]			
	Mean	Stdev	Min	Max
Gaudet[7]	0.01131	0.01341	0.00052	0.14259
LSPI1 Controller	0.14614	0.07574	0.00471	0.73722
DPS1 Controller	0.13811	0.30381	0.00023	3.81184
DPS2 Controller	0.01556	0.02470	0.00009	0.36810

Table 4: Performance comparison between previous work and the newly obtained controllers from two different RL methods.

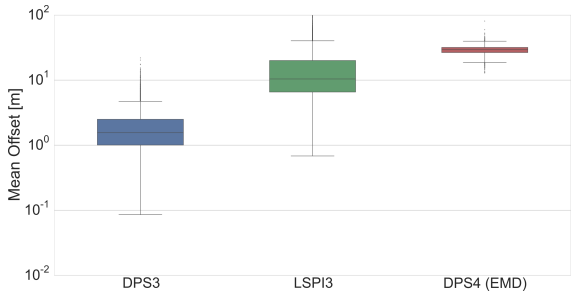

Figure 7: Post evaluation of the task where the controllers are not provided with the relative position information.

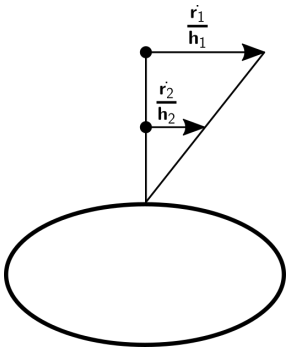

Figure 8: The "gain problem". The EMD input to the controller looks the same for two different scenarios, i.e., $\frac{\dot{r}_1}{h_1} = \frac{\dot{r}_2}{h_2}$.

an offset to the target position, the transient response contains only the initial velocity canceling. During the training phase, controllers were exposed to the environments for about 10000 seconds. It turns out that the controllers start to chatter the lower their spacecraft's mass becomes. This could be because the controllers were never exposed to this scenario of having low mass during training. We therefore cut the graphs after 10000 seconds and focus on the performance during that part. As shown in Figure 7, the DPS3 controller manages to keep the drift on average in less than 3 meters during one hour. This is our baseline for the EMD controller.

It turned out that the EMD controller works only in a more restricted environment. Testing the controller in the full environment range in terms of the scaling factor s used in Algorithm 2 resulted in average offsets to the starting point of several hundreds of meters. Therefore, we considered in the end a very limited range of altitudes by setting the scaling factor $s = 1.1$. This way, the EMD controller stays on average within 81 meters in the worst case and drifts on average around 15 times more than our baseline.

The controller has what we call a "gain problem". There exist infinite scenarios with different heights and velocities for which the sensory input to the controller looks the same. Being very distant to the surface with high relative velocity and thus command a high thrust is a good action. However, being very close to the surface with very low relative velocity and giving the same high thrust command has a very different effect and is a bad action. The policy architecture however can not cope with this fact, as it learns the best action for a given state, i.e., sensory input. This causes the controller to use about 25 times more propellant than an optimal nominal acceleration controller would use. However, the worst case scenario shows that our EMD controller is able to hover above a small body with some drift

	Average Offset to Starting Position [m]			
	Mean	Stdev	Min	Max
DPS3 Controller	2.03480	1.64498	0.08570	21.96248
LSPI3 Controller	18.52626	20.29895	0.67044	226.89031
DPS4 (EMD) Controller	29.16655	4.24756	12.88770	80.82599

Table 5: Performance of the second two experiments where the controllers are not provided with the relative position information.

for a short period of time. Based on the current findings using RL, it boils down to a tradeoff of whether the benefits of an EMD system as mentioned in Section B are worth the much higher fuel consumption caused by it. A more suitable usage of the RL EMD controller could be a lightweight, low power emergency backup navigation system.

B. Discussion

We obtained with our evolutionary direct policy search a significant better hovering controller compared to previous work.[7] There, the hovering task was decoupled into three sub tasks of driving the offset in each dimension to zero, and an independent policy was learned for each dimension. The MDP simulator used here is slightly different than in the previous setup as there is for example no delay between the controller issuing the command and the thruster actually performing the command. However, in contrast to the previous work's control frequency of 10Hz, we use a control frequency of 1Hz which compensates for this difference. Furthermore, the small body model implemented here is more general and intuitively makes the learning problem more difficult. The policy learned here is allowed to fuse information of all dimensions to command optimal control actions and no human pre-knowledge about the problem has been introduced into the policy architecture. Although the generational DPS is required to search in a larger parameter space, it obtained a better policy resulting in a controller with an average steady state target offset error of about 5mm (Table 4).

We showed that is is possible to hover with some drift using only EMD sensory input in a restricted environment (Table 5). Since the sensory system for the EMD controller does not measure altitude an issue with this controller is the "gain problem", i.e., the controller cannot distinguish between high speed and high altitude and low speed and low altitude. The knowledge of the spacecraft's height is very powerful. An experiment based on a pure velocity sensory input (Table 5) improved the hovering performance by an order of magnitude in terms of average target position offset and fuel consumption. Further experiments could combine height measurements with EMD sensory input to see if the gain problem vanishes. At this point, we conclude that a pure EMD controller developed with RL could be used as a supplementary lightweight emergency system in case the main telemetry system fails. Changing the policy architecture might improve the EMD controller. A FFNN was chosen as the policy architecture for the DPS algorithm. One could have also considered other, more powerful architectures which would have memory capabilities such as recurrent neural networks or long short-term neural networks.[35] However, changing the architecture can lead to computational problems. The more parameters the architecture comprises, the larger the DPS search space is. Thus, evolutionary based optimization will require larger populations or more generations to obtain solutions. Generations though are very expensive in time. Another problem from the learning perspective is the higher chance of overfitting for architectures with large parameter spaces.

The low control frequency was chosen because of the necessity of having a fast fitness evaluation, i.e., computing the utility which requires $k = 10$ simulations of about 9000 simulation seconds. Measurements showed that one computation of the utility function takes about six seconds. If one considers a population of 1000 individuals, evolving one generation on a nonparallel architecture takes more than an hour. So either the number of generations has to be kept low or the population size has to be reduced. The fixed control frequency is a core obstacle which needs to be redesigned. If one allows a dynamic control frequency and assumes perfect sensor measurements (i.e., no sensory noise model), an adaptive integrator will improve the evaluation performance and thus allow more experiments in less time. Our RL framework has a modular architecture which allows to use richer models for both, spacecraft and small body. For example, a polyhedral implementation of the small body can model a much wider range of shapes than ellipsoids. The more flexible model would allow to reflect depressions, ridges, cliffs, caverns, and holes. The spacecraft is currently modeled as a 3DOF point mass and could be extended to a 6DOF rigid body model.

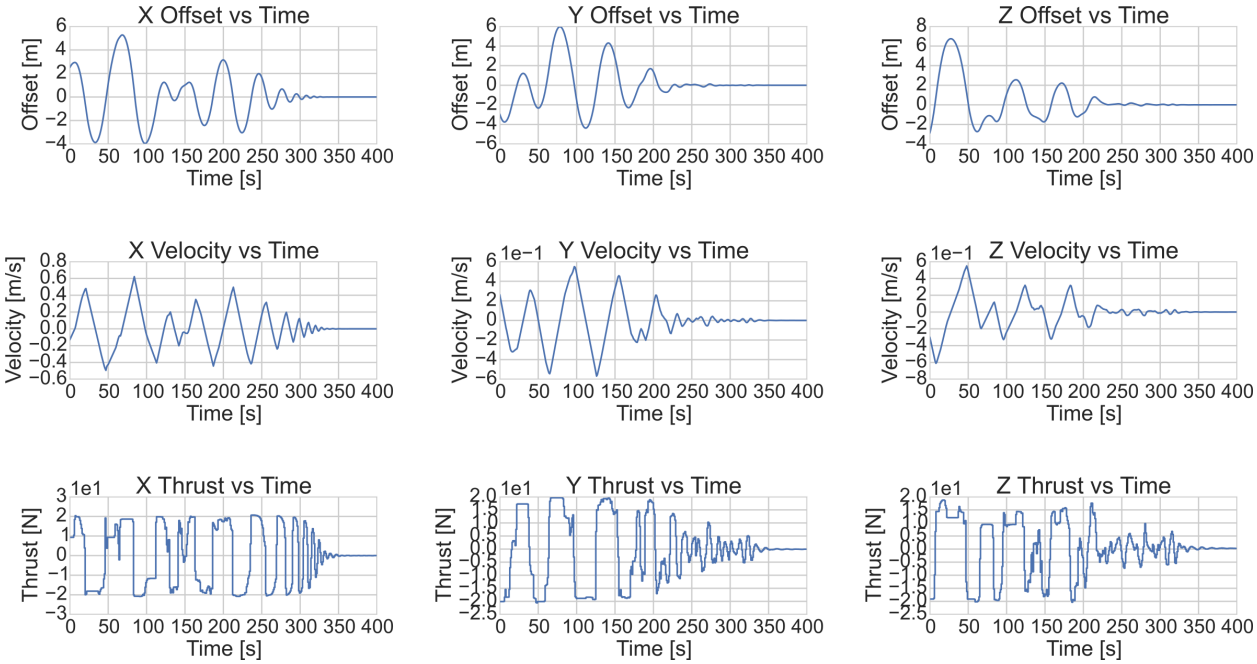

Figure 9: Transient response for the worst case scenario of the DPS1 controller. The controller uses its full range of thrust and reaches steady state after 350 seconds.

VII. Conclusions

We model the control problem of a spacecraft orbiting in a unknown gravity field originated from a tumbling small body as a Markov Decision Problem and use a Direct Policy Search algorithm to find control policies able to keep the spacecraft hovering on a designated point. Unlike previous work we consider a triaxial ellipsoid and experiment with a number of different sensory inputs. We find our approach to be able to deliver policies able to minimize drift also when elementary motion detectors are considered as the only proprioceptive sensor on board. Our work is a first step towards obtaining landing algorithms based on visual clues and working in low-gravity environments.

References

[1]B. Chauvineau, P. Farinella, and F. Mignard, "Planar Orbits about a Triaxial Body: Application to Asteroidal Satellites," *Icarus*, Vol. 105, No. 2, 1993, pp. 370–384.

[2]D. Scheeres, B. Williams, and J. Miller, "Evaluation of the Dynamic Environment of an Asteroid: Applications to 433 Eros," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 3, 2000, pp. 466–475.

[3]S. Sawai, D. Scheeres, and S. Broschart, "Control of Hovering Spacecraft Using Altimetry," *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 4, 2002, pp. 786–795.

[4]D. Scheeres, "Orbital mechanics about small bodies," *Acta Astronautica*, Vol. 72, 2012, pp. 1–14.

[5]S. B. Broschart and D. Scheeres, "Control of Hovering Spacecraft Near Small Bodies: Application to Asteroid 25143 Itokawa," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 2, 2005, pp. 343–354.

[6]R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*. MIT Press, 1998.

[7]B. Gaudet and R. Furfaro, "Robust Spacecraft Hovering Near Small Bodies in Environments with Unknown Dynamics using Reinforcement Learning," *AIAA/AAS Astrodynamics Specialist Conference*, 2012.

[8]D. Izzo, N. Weiss, and T. Seidl, "Constant-Optic-Flow Lunar Landing: Optimality and Guidance," *Journal of Guidance, Control, and Dynamics*, Vol. 34, No. 5, 2011, pp. 1383–1395.

[9]P. Zarchan, "Tactical and Strategic Missile Guidance Fourth Edition," *Progress in Astronautics and Aeronautics*, Vol. 199, 2002.

[10]D. Simon, "Kalman filtering," *Embedded Systems Programming*, Vol. 14, No. 6, 2001, pp. 72–79.

[11]C. Mead, "Neuromorphic Electronic Systems," *Proceedings of the IEEE*, Vol. 78, No. 10, 1990, pp. 1629–1636.

[12]G. Indiveri, "Analog VLSI Model of Locust DCMD Neuron Response for Computation of Object Approach," *Progress in Neural Processing*, Vol. 10, 1998, pp. 47–60.

[13]T. Netter and N. Francheschini, "A Robotic Aircraft that Follows Terrain Using a Neuromorphic Eye," *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, Vol. 1, IEEE, 2002, pp. 129–134.

[14]F. Ruffier and N. Franceschini, "OCTAVE: a bioinspired visuo-motor control system for the guidance of Micro-Air-Vehicles," *Microtechnologies for the New Millennium 2003*, International Society for Optics and Photonics, 2003, pp. 1–12.

[15]F. Ruffier and N. Franceschini, "Optic flow regulation: the key to aircraft automatic guidance," *Robotics and Autonomous Systems*, Vol. 50, No. 4, 2005, pp. 177–194.

[16]S. Thakoor, J. Chahl, M. V. Srinivasan, L. Young, F. Werblin, B. Hine, and S. Zornetzer, "Bioinspired engineering of exploration systems for NASA and DoD," *Artificial Life*, Vol. 8, No. 4, 2002, pp. 357–369.

[17]V. Medici, G. Orchard, S. Ammann, G. Indiveri, and S. Fry, "Neuromorphic computation of optic flow data," tech. rep., Technical Report-Ariadna 08/6303, European Space Agency, the Advanced Concepts Team, 2010.

[18]F. Valette, F. Ruffier, S. Viollet, and T. Seidl, "Biomimetic optic flow sensing applied to a lunar landing scenario," *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, IEEE, 2010, pp. 2253–2260.

[19]S. Parkes, I. Martin, M. Dunstan, and D. Matthews, "Planet Surface Simulation with Pangu," *Eighth International Conference on Space Operations*, 2004, pp. 1–10.

[20]D. Izzo and G. D. Croon, "Landing with time-to-contact and ventral optic flow estimates," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 4, 2012, pp. 1362–1367.

[21]L. Landau and E. Lifshitz, "Mechanics, vol. 1," *Course of Theoretical Physics*, 1976.

[22]J. Ivory, "On the Attractions of homogeneous Ellipsoids," *Philosophical Transactions of the Royal Society of London*, Vol. 99, 1809, pp. 345–372.

[23]W. D. MacMillan, "The Theory of the Potential," 1958.

[24]J. Danby, "Fundamentals of Celestial Mechanics," *Richmond: Willman-Bell,— c1992, 2nd ed.*, Vol. 1, 1992.

[25]F. R. Moulton, *An Introduction to Celestial Mechanics*. Courier Corporation, 2012.

[26]D. O. Cersosimo, *Evaluation of Novel Hovering Strategies to Improve Gravity-tractor Deflection Merits*. PhD thesis, University of Missouri–Columbia, 2011.

[27]B. Gaudet and R. Furfaro, "Adaptive Pinpoint and Fuel Efficient Mars Landing Using Reinforcement Learning," *Automatica Sinica, IEEE/CAA Journal of*, Vol. 1, No. 4, 2014, pp. 397–411.

[28]D. Izzo, L. F. Simões, and G. De Croon, "An evolutionary robotics approach for the distributed control of satellite formations," *Evolutionary Intelligence*, Vol. 7, No. 2, 2014, pp. 107–118.

[29]J. Kennedy, "Particle Swarm Optimization," *Encyclopedia of Machine Learning*, pp. 760–766, Springer, 2010.

[30]V. Heidrich-Meisner and C. Igel, "Evolution Strategies for Direct Policy Search," *Parallel Problem Solving from Nature–PPSN X*, pp. 428–437, Springer, 2008.

[31]J. Koutník, G. Cuccu, J. Schmidhuber, and F. Gomez, "Evolving Large-Scale Neural Networks for Vision-Based Reinforcement Learning," *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, ACM, 2013, pp. 1061–1068.

[32]J. Koutník, J. Schmidhuber, and F. Gomez, "Evolving Deep Unsupervised Convolutional Networks for Vision-Based Reinforcement Learning," *Proceedings of the 2014 conference on Genetic and evolutionary computation*, ACM, 2014, pp. 541–548.

[33]D. Izzo, M. Ruciński, and F. Biscani, "The Generalized Island Model," *Parallel Architectures and Bioinspired Algorithms*, pp. 151–169, Springer, 2012.

[34]M. G. Lagoudakis and R. Parr, "Least-Squares Policy Iteration," *Journal of Machine Learning Research*, Vol. 4, 2003, pp. 1107–1149.

[35]S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, Vol. 9, No. 8, 1997, pp. 1735–1780.