aerospace

MDPI

Article

Safe Motion Planning and Learning for Unmanned Aerial Systems

Baris Eren Perk [1,*] and Gokhan Inalhan [1,2]

[1] Faculty of Aeronautics and Astronautics, Istanbul Technical University, Istanbul 34469, Turkey
[2] School of Aerospace, Transport & Manufacturing, Cranfield University, Bedfordshire MK43 0AL, UK; inalhan@cranfield.ac.uk
* Correspondence: perkb@itu.edu.tr

Abstract: To control unmanned aerial systems, we rarely have a perfect system model. Safe and aggressive planning is also challenging for nonlinear and under-actuated systems. Expert pilots, however, demonstrate maneuvers that are deemed at the edge of plane envelope. Inspired by biological systems, in this paper, we introduce a framework that leverages methods in the field of control theory and reinforcement learning to generate feasible, possibly aggressive, trajectories. For the control policies, Dynamic Movement Primitives (DMPs) imitate pilot-induced primitives, and DMPs are combined in parallel to generate trajectories to reach original or different goal points. The stability properties of DMPs and their overall systems are analyzed using contraction theory. For reinforcement learning, Policy Improvement with Path Integrals (PI^2) was used for the maneuvers. The results in this paper show that PI^2 updated policies are a feasible and parallel combination of different updated primitives transfer the learning in the contraction regions. Our proposed methodology can be used to imitate, reshape, and improve feasible, possibly aggressive, maneuvers. In addition, we can exploit trajectories generated by optimization methods, such as Model Predictive Control (MPC), and a library of maneuvers can be instantly generated. For application, 3-DOF (degrees of freedom) Helicopter and 2D-UAV (unmanned aerial vehicle) models are utilized to demonstrate the main results.

Keywords: UAV; artificial intelligence; contraction theory; nonlinear control; primitives; reinforcement learning; imitation learning; maneuvers

Citation: Perk, B.E.; Inalhan, G. Safe Motion Planning and Learning for Unmanned Aerial Systems. *Aerospace* **2022**, *9*, 56. https://doi.org/10.3390/aerospace9020056

Academic Editor: Joost Ellerbroek

Received: 21 December 2021
Accepted: 17 January 2022
Published: 22 January 2022

1. Introduction

Unmanned aerial systems have gained significant importance in the last few decades. Inherently, they are more agile and more suitable for working in dangerous areas in comparison with manned systems. Accordingly, optimization and learning techniques are employed in their applications to increase the number of UAV types and extend their operation range. In [1], centralized path planning based on reinforcement learning is implemented in a combat aerial vehicle fleet in order to avoid enemy defense systems. To localize radio frequency emitting targets in the operation area, multiple UAVs were deployed in [2] using Particle Filter and Extended Kalman Filter algorithms and vision-based detection. For the design optimization and mathematical modeling of unmanned aerial vehicles, a nonlinear lifting line method is proposed in [3]. This new modeling methodology is applied on the Aerospace Research Center (ARC) UAV in order to demonstrate that faster and cheaper prototyping is possible for future UAV models.

Despite enormous advancements in learning and optimization research, generating safe and aggressive movements is a rather hard problem for engineers to solve. Actuator saturation and nonlinear and nonholonomic dynamics are key issues. Most of the applications of UAVs, such as agriculture, intelligence, surveillance, and reconnaissance, only require trimmed flight. However, humans, as well as animals, are experts in generating

seemingly hard but stable trajectories despite environmental and dynamic challenges. In acrobatic maneuvers, expert pilots demonstrate maneuvers that are deemed at the edge of the flight envelope.

To generate feasible maneuvers, on the other hand, pilot-executed trajectories are used in [4] to develop nonlinear dynamic models of helicopters. Using this model, intuitive control strategies derived from pilot behaviors are demonstrated in a hardware-in-the-loop simulation environment. Multiple expert executed flight demonstrations are also used in [5] to generate a desired trajectory. Additionally, a receding horizon differential dynamic programming controller is employed for a local model of the helicopter, which is learned from previous flight demonstrations. In [6,7], optimization methods are first employed and then performance is enhanced iteratively through experimentation. In recent years, nonlinear controller designs [8–11] have also been employed to generate safe maneuvers. For safe path planning [12–15], on the other hand, cost functions are utilized to minimize the flight envelope. Under these approaches, however, conservative maneuvers are generated in comparison with pilot-induced trajectories.

In general, choosing an optimization/learning methodology is only one aspect of the problem. In many scenarios, it is crucial to guarantee the safety of the system while learning desired behaviors. As a possible solution, Lyapunov-based methods [16–18] were proposed to solve reinforcement learning problems. However, there is no common rule for finding a suitable Lyapunov function and using Lyapunov functions; if any such function exists, it does not necessarily result in aggressive maneuvers.

For optimization problems, the stability of the receding horizon scheme can be guaranteed by adding a terminal cost function consisting of a Control Lyapunov Function (CLF) or by applying a sufficient prediction horizon without a terminal cost function, where a general cost function qualifies as Lyapunov function [19]. In [20], Nonlinear Model Predictive Control is applied to a nonholonomic mobile robot with a kinematic model for path following. The stability of the system is guaranteed by appropriately choosing a prediction horizon [21]. As shown in [20], the maneuvers were carefully selected to find the prediction horizon. Similar to reinforcement learning applications, finding a CLF in optimal control problems is not an easy task, and CLF eliminates other possible, may be more effective, solutions. Therefore, such methods lead to conservative solutions in order to guarantee stability.

For biological systems, on the other hand, one can naively guess that learning is achieved by following two sequential steps. First, the movement is imitated, and, next, the imitated behavior is improved through trial and error. It was observed in humans pilots that maneuvers performed with remote helicopters are often repeatable [4]. Each specific maneuver type is predictable in its own sequencing and duration, which suggests a primitive structure. In animals, experiments also show that kittens that have been exposed to adult cats retrieve food much faster in comparison to the control group [22]. Moreover, experiments on monkeys [23,24] also suggest that movements follow a virtual trajectory, which is independent of initial conditions and closely related to feedback control. This finding is also supported by other experiments conducted on frogs [25,26].

The most interesting discovery in the experiments concerning frogs was that the fields generated by activation on the spinal cord follow a principle vectorial summation. Simultaneous stimulation of two different areas of the spinal cord resembles the vectorial summation of the separate activation of each of these areas (See Figure 1) [27]. Generally, it is found that simultaneous stimulation fields and vectorial summations were similar in 87% of the experiments. As mentioned in [28], the vector summation of force fields suggests that nonlinear forms of interactions among neurons and between neurons and muscles is more or less eliminated. With few force fields stored in the spinal cord, one may represent motor primitives using the superposition of spinal fields:

$$D(q, \dot{q}, \ddot{q}) = u \sum_{i=1}^{K} \alpha_i \rho_i(q, \dot{q}, \ddot{q}), \tag{1}$$

where D and u represent system dynamics and force, respectively. In this equation, force fields (ρ_i) are weighted by scalar and positive coefficients, (α_i). In [28], it is also suggested that these spinal fields are selected using supraspinal signals by evaluating how much each field contributes to the overall field.

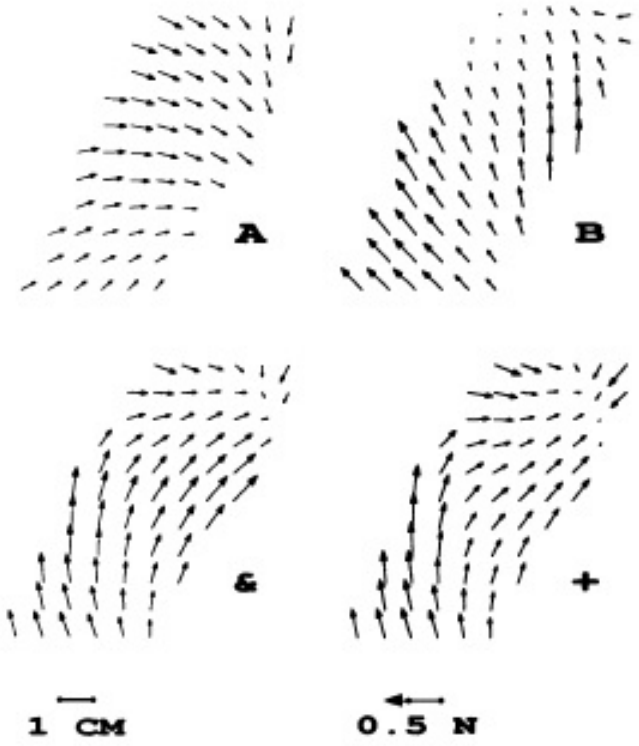

Figure 1. Costimulation fields (&) and summation fields (+) of (**A**) and (**B**) [27].

As a result, this study is focused on learning about and improving feasible trajectories inspired by biological systems. We introduce a framework that leverages methods in the field of control theory and reinforcement learning. We show that we are in the stable flight regime. In practice, inner-loop nonlinear controllers can mainly be used to avoid unmodeled dynamics and disturbances. For primitive structures, DMPs [29] are selected to imitate such primitives of motion. In DMPs, the kinematic trajectories (i.e., positions, velocities, and accelerations) of the motion are generated and converted to motor commands using feedforward controllers, probably by an inverse dynamic model and stabilized by feedback controllers. It is assumed that other non-linearities shall be handled by the controller at the motor execution phase. Detailed definitions of the DMPs can be found in Appendix A. In our study, movement primitives are also combined in parallel to generate feasible trajectories to reach original or different goal points.

Instead of constructing Lyapunov functions for stability, we applied contraction analysis [30] for DMPs, as well as for the reinforcement learning. In general, contraction is a form of stability where all trajectories with different initial conditions contract into a single trajectory. As mentioned in the discussion of previous experiments above, this form of stability is also implied in the structure of virtual trajectories. Additionally, we believe that contraction analysis should be studied for learning, as it was hypothesized in [31] that humans generate stable trajectories so that perturbations have less effect and require little correction; as a result, humans exploit contraction regions.

For reinforcement learning, PI^2 [32] is used. In PI^2, trajectories are updated using stochastic Hamilton–Jacobi–Bellman (HJB) equations and direct policies are learned by approximating a path integral, in which the statistical inference problem is solved using sample roll-outs. Only one open parameter, exploration noise, is required in this learning

method, and updates are not limited by numerical instabilities, since neither matrix inversions nor gradient learning rates are needed. As a result, it is shown that the performance of PI^2 is significantly better than that of gradient-based policy learning, and it can be applied in high-dimensional problems [32].

Although, there are successful studies using pilot-induced trajectories for UAVs, our goal in this research is to design a stable biologically inspired framework, which can be used to imitate and regenerate feasible, possibly aggressive, trajectories. The parallel combination of dynamic movement primitives coupled with reinforcement learning is firstly introduced for motion planning, and contraction analysis is used for the stability analysis. In general, the contribution of this paper is three-fold. First, while DMPs with contraction analysis [33] and PI^2 with DMPs [32] have been studied before, our proposed methodology in Section 2 encompasses all three methods to generate biologically inspired, feasible and possibly aggressive maneuvers. Second, DMPs are combined in parallel to guarantee stability. Third, it is shown that PI^2 updated policies are feasible and parallel combination of different updated primitives transfer the learning in contraction regions.

The remainder of the paper is structured as follows. In Section 2, we introduce the methodology and make several remarks for our study. First, DMP can be directly used in controller design, where additional feedback controllers as proposed in [34] are not required. Second, offline learning can be performed using feasible trajectories. Third, having library of motion primitives, one can combine primitives instantly in order to generate feasible trajectories in volatile environments. In Section 3 we demonstrate our results on 3-DOF helicopter and 2D UAV models.

2. Methodology

As mentioned in Section 1, our approach mainly encompasses the methods listed below:

- Contraction Theory.
- Dynamic Movement Primitives.
- PI^2 Reinforcement Learning Algorithm.

In our methodology, firstly, maneuvers generated by human operators or optimization programs such as MPC are imitated by DMPs. These primitives are then combined in parallel in such a way that the resultant primitive is also contracting. Subsequently, PI^2 is applied to improve the performance of the combined primitive. For the methodology, we will analyze the compatibility and applicability of these methods and theories below.

2.1. Contraction Analysis of Systems with DMPs

For a system defined below

$$\dot{x} = f(x, t), \tag{2}$$

the system is contracting [30], if the infinitesimal displacement at a fixed time:

$$\delta\dot{x} = \frac{\partial f(x,t)}{\partial x}\,\delta x, \tag{3}$$

decays over time. Therefore, the Jacobian $\frac{\partial f}{\partial x}$ should be a negative definite in that region.

Similar to linearizing the system at equilibrium points, we can use Jacobian to study the convergence of trajectories irrespective of initial conditions. This requirement, however, is a sufficient, but not a necessary, condition. One can define $\delta z = \Theta\delta x$, where Θ is a square matrix, and length is defined $\delta z^T\delta z = \delta x^T M\delta x$, where M is a metric. Hence, the negative definitiveness of a general Jacobian, $F = (\dot{\Theta} + \Theta\frac{\partial f}{\partial x})\Theta^{-1}$, is a necessary and sufficient condition for the system to contract [30].

In [33], it is demonstrated that discrete DMPs contract in hierarchy. As DMPs are learned from feasible trajectories, our goal is to study the stability of the system. In other

words, we aim to find out if the overall system will contract to the desired trajectories. For the analysis, the system differential equation can be defined in the following form:

$$\dot{x} = f(x) + B(x)u \tag{4}$$

Then, virtual displacement dynamics is defined as:

$$\delta\dot{x}(t) = \frac{\partial(f(x) + B(x)u)}{\partial x}\delta x + B(x)\delta u(t). \tag{5}$$

In a scalar case where $Bu = -f(x_{ref}) + K_p(x_{ref} - x)$, the virtual displacement dynamics is reformulated as:

$$\delta\dot{x}(t) = \frac{\partial\big(f(x) - K_p x\big)}{\partial x}\delta x + \frac{\partial\Big(K_p x_{ref} - f(x_{ref})\Big)}{\partial x_{ref}}\delta x_{ref}. \tag{6}$$

If the reference system is contracting, its effect will be bounded and will decay [30] as $\delta x_{ref} \to 0$, and the system will contract in hierarchy with a proper choice of K_p, so that $\frac{\partial\left(f(x) - K_p x\right)}{\partial x}$ is uniformly negative. This is analogous to pole placement in linear control theory. Overall, if the system is designed to contract and a particular solution of the system is x_{ref}, the system will converge to the reference trajectory. In general, the stability of these derived feasible trajectories can be studied using several methods in contraction theory, as summarized in Appendix B.

For DMPs, let's assume the standard manipulator equation [35]:

$$\ddot{q} = H(q)^{-1}(C(q,\dot{q})\dot{q} + g(q) + u) \tag{7}$$

$$u = -g(q) + K_p(q_d - q) + K_d(\dot{q}_d - \dot{q}) \tag{8}$$

$$\ddot{q}_d = D(q_d, \dot{q}_d), \tag{9}$$

where $H(q)$ is a positive definite inertia matrix, C represents the Coriolis and centripetal forces, g is the force of gravity, q is the joint angle, and u is forces/torques. D also stands for the DMP equations. Since DMP is contracting, its effect on Equation (7) will be bounded, and DMP will act as an exponential decaying disturbance to the original system. As a result, the system contracts to a particular solution with appropriate choices of K.

For the combination of primitives defined in Appendix B, let's again take the standard manipulator equation with DMPs. If any two or more primitives are contracting, then the linear combination of the reference system in the system dynamics is also contracting such that:

$$\ddot{q} = H(q)^{-1}(C(q,\dot{q})\dot{q} - K_p q - K_d\dot{q} + \sum_{i=1}^{n}\alpha_i(K_p q_{d_i} + K_d\dot{q}_{d_i}), \tag{10}$$

where $\sum_{i=1}^{n}\alpha_i = 1$. It is important to note that, if there is no singularity, any combination of these torques will be within the limits of the torque constraints.

We can combine primitives for the same or different goals as shown in Figure 2. In both approaches, the areas in between primitives are reachable. It is possible to combine primitives that generate trajectories that pass through these areas.

One can observe that the combination of system differential equations necessitates the combination of primitives. As a result, only the controlled section of differential equations is modified in Equation (10). Since DMP weights are designed linearly, a simple combination of weights is enough to achieve the desired results. In addition, primitives can also be combined serially. It was shown in [33] that obstacle avoidance maneuvers can be divided into primitives. For this type of combination, it is also possible to modify DMPs using their own parameters.

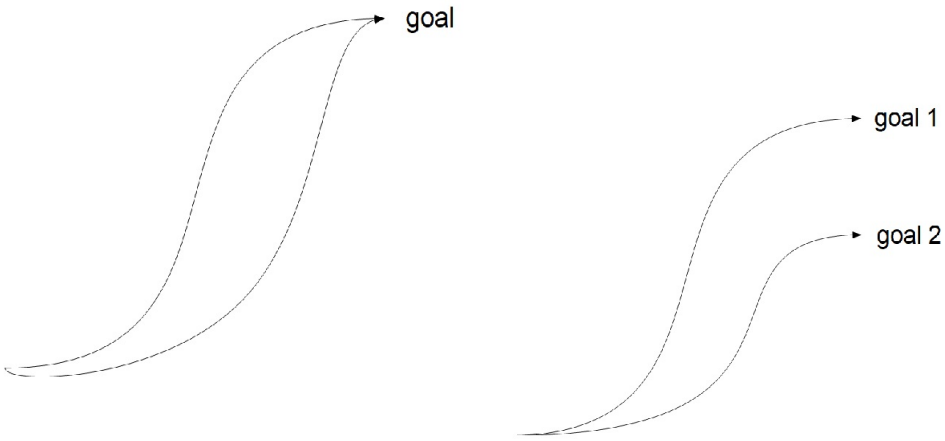

Figure 2. Primitive Combinations.

2.2. Using Reinforcement Learning with Contraction Analysis

In the last few decades, Reinforcement learning has attracted attention as a learning method for studying movement planning and control [36]. Reinforcement learning is a concept that is based on trial and error, typically the constant evaluation of performance in a surrounding environment is also required. In general, reinforcement learning requires an unambiguous representation of states and actions, as well as a scalar reward function.

Reinforcement learning is simple in its form, but it is mostly regarded as impractical due to the curse of dimensionality. Since value function approaches remain problematic in high dimensions, direct policy learning is utilized as an alternative method. Even for direct policy learning, however, numerical issues and the handling of different parameters still need to be addressed.

To compensate for such issues, a new methodology of probabilistic learning was derived on the basis of stochastic optimal control and path integrals. Policy Improvement with Path Integrals (PI^2) [32] connects between value function approximation using the stochastic HJB equations and direct policy learning by approximating a path integral, that is, by solving a statistical inference problem from sample roll-outs. The resulting algorithm is numerically robust, and only one parameter, exploration noise, is needed for its application.

Such methodology was first applied in stochastic optimal control. For optimal control, the cost functions are defined as:

$$R(\tau_i) = \omega_{t_N} + \int_{t_i}^{t_N} r_t dt \tag{11}$$

$$r_t = r(x_t, u_t, t) = q_t + \frac{1}{2} u_t^T R u_t, \tag{12}$$

where $\omega_{t_N} = \omega(x_{t_N})$ is a terminal cost at time t_N, and r_t denotes the immediate cost at time t. Moreover, $q(t)$ and R represent the immediate cost at time t and the semi-definite weight matrix of control cost, respectively.

For a general system,

$$\dot{\mathbf{x}}_\mathbf{t} = \mathbf{f}(\mathbf{x}_t, t) + \mathbf{G}(\mathbf{x}_t)(\mathbf{u}_t + \varepsilon_t) = \mathbf{f}_t + \mathbf{G}_t(\mathbf{u}_t + \varepsilon_t), \tag{13}$$

we aimed to find a control input, $\mathbf{u_t}$, which minimized the value function:

$$V(\mathbf{x}_{t_i}) = V_{t_i} = \min_{\mathbf{u}_{t_i:t_N}} E_{\tau_i}[R(\tau_i)], \tag{14}$$

where E_{τ_i} is (τ_i) for all trajectories. The stochastic HJB equation [37,38] for the optimal control problem is shown below:

$$-\partial_t V_t = \min_{\mathbf{u}}(r_t + (\nabla_x V_t)^T + \mathbf{F}_t + \frac{1}{2} trace((\nabla_{xx} V_t)\mathbf{G}_t \Sigma_\varepsilon \mathbf{G}_t^T)), \tag{15}$$

where $\mathbf{F}_t = \mathbf{f}(\mathbf{x}_t, t) + \mathbf{G}(\mathbf{x}_t)\mathbf{u}_t$.

In [32], optimal control is studied extensively. Subsequently, their study was extended to reinforcement learning, in which a system model is not used. In their approach, the desired state is taken as an input, as action is viewed as any input to the control system. For such a control structure:

$$\ddot{q} = H(q)^{-1}(-C(q, \dot{q}) - v(q)) + H(q)^{-1}u \tag{16}$$

$$u = K_p(q_d - q) + K_D(\dot{q}_d - \dot{q}) \tag{17}$$

$$\ddot{q}_d = G(q_d, \dot{q}_d)(\theta + \varepsilon_{t_i}). \tag{18}$$

Here, Equation (18) generates the desired trajectories. Since only the controlled differential equation is used in path integral formulation, only Equation (18) is needed to apply reinforcement learning. Uncontrolled sections of the system dynamics are not required.

Applying the PI^2 algorithm to the controlled section of DMPs, the equations of the algorithm [32] are formulated below:

$$S(\tau_{i,k}) = \omega_{t_N,k} + \Sigma_{j=i}^{N-1} q_{t_j,k} + \frac{1}{2}\Sigma_{j=i+1}^{N-1}(\theta + \mathbf{M}_{t_j,k}\varepsilon_{t_j,k})^T \mathbf{R}(\theta + \mathbf{M}_{t_j,k}\varepsilon_{t_j,k}) \tag{19}$$

$$\mathbf{M}_{t_j,k} = \frac{\mathbf{R}^{-1} g_{t_j,k} g_{t_j,k}^T}{g_{t_j,k}^T \mathbf{R}^{-1} g_{t_j,k}} \tag{20}$$

$$P(\tau_{i,k}) = \frac{e^{-\frac{1}{\lambda} S\tau_{i,k}}}{\Sigma_{k=1}^{K} e^{-\frac{1}{\lambda} S\tau_{i,k}}} \tag{21}$$

$$\delta\theta_{t_i} = \Sigma_{k=1}^{K}[P(\tau_{i,k})\mathbf{M}_{t_i,k}\varepsilon_{t_i,k}] \tag{22}$$

$$\delta\theta_j = \frac{\Sigma_{i=0}^{N-1}(N-i)w_{j,t_i}[\delta\theta_{t_i}]_j}{\Sigma_{i=0}^{N-1}(N-i)w_{j,t_i}} \tag{23}$$

$$\theta^{(new)} = \theta^{(old)} + \delta\theta \tag{24}$$

Under this setup, Equation (19) computes the cost function at each time step, i, for each trial, k, in the epoch. Equation (20) represents the projection matrix onto the $g_{j,k}$, i.e., the basis function from the system dynamics, under the metric R^{-1}. Equation (21) determined the probability of each trial. We can simply infer from $P(\tau_{i,k})$ that lower cost trajectories have higher probabilities. Equation (22) determines the parameter update at each time step. It can easily be seen that trajectories with lower cost contribute more to the parameter update. Equation (23) averages the parameter update, $\delta\theta_{t_i}$, of each time step. Finally, Equation (24) updates the old parameter.

Using this formulation, we aim to understand the stability aspect of the system, and whether the system contracts from the initial conditions. It was already shown in [33] that discrete DMPs contract. Using hierarchical and other properties of contraction theory, our goal is to find contracting regions for the system setup. One may again find that the probability distribution of $P(\tau_{i,k})$ serves as a tool for the parallel combination of inputs while updating the desired trajectories. If each roll-out is contracting, we can conclude that the updated trajectories will also be contracting. This is analogous to creating an envelope as shown in Figure 2. We can use controllers, such as the one defined in Equation (44), in order to generate contracting trajectories and apply them to reinforcement learning.

Moreover, each learned DMP is combined in order to generate a new primitive with a different goal point. Such combination is shown below:

$$\begin{aligned}\dot{y}_d =& \alpha_1(f(g_1, y_t, z_t) + g\theta_1) + \cdots + \alpha_n(f(g_n, y_t, z_t) + g\theta_n) \\ =& f(\sum_{i=1}^{n} \alpha_i g_i, y_t, z_t) + g\sum_{i=1}^{n} \alpha_i \theta_i,\end{aligned} \tag{25}$$

where $\sum_{i=1}^{n} \alpha_i = 1$ and $\alpha_i > 0$. Since the new primitive is contracting, we can further continue learning using PI^2 updates for this primitive. In the PI^2 learning algorithm (Equation (22)), we see that each parameter, θ_{t_i}'s, for K roll-outs is updated as shown below:

$$\delta\theta_{t_i} = \Sigma_{k=1}^{K}[P(\tau_{i,k})\mathbf{M}_{t_i,k}\varepsilon_{t_i,k}]. \tag{26}$$

Here, probability of the negative cost function is used to weight only the open parameter, i.e., the exploration noise ε. We assume that the system is contracting with feasible DMPs. Thus, the system always reaches a goal point at the end of the trajectory, and the cost function $\omega_{t_N,k}$ in $S(\tau_{i,k})$ can safely beignored. We also assume that q's will contract to the desired trajectory q_d's. Therefore, in a contracting region, we define q's as a combination of θ and ε's, which the only parameters to be used in the cost function $S(\tau_{i,k})$ as shown below:

$$S(\tau_{i,k}) = \Sigma_{j=i}^{N-1} q_{d_j,k} + \frac{1}{2}\Sigma_{j=i+1}^{N-1}(\theta + \mathbf{M}_{t_j,k}\varepsilon_{t_j,k})^T\mathbf{R}(\theta + \mathbf{M}_{t_j,k}\varepsilon_{t_j,k}). \tag{27}$$

At each update sequence in a contraction region, the primitive updates are weighted with θs. For the combination of primitives (Equation (25)), on the other hand, updates are weighted with αs and θs, and the probability functions are taken as the average of the probability functions of separate primitives, if we assume that we use the same exploration noise, ϵ, for all primitives. Similarly, when we update primitives separately and combine them, we also average the probability functions. As a result, the learning curve of a combined primitive would be improved even further as the primitive learning process continues.

2.3. Remarks

Generally, contraction analysis yields simpler results which can be directly applied to systems/methodologies. In this section, we make a few application remarks, in which contraction analysis is used in combination with DMPs and possible learning methods.

2.3.1. DMP as a Controller

DMPs were originally used in a combination of feedback and feedforward controllers. However, it can be deduced from contraction analysis that one can directly use a DMP in a controller. Consider a system:

$$\dot{x} = f(x) + bu, \tag{28}$$

where we learn DMPs, and a controller, u, can be generated as $u = \frac{\dot{x}_d - f(x_d)}{b}$. In the closed loop system, particular solution of the system is $x = x_d$. Since DMPs are contracting, and the DMP parameters act as PD terms for the overall system, contraction to movement primitives is guaranteed. This controller can be regarded as a system with added virtual springs and dampers, and the path-to-goal point is modified by the basis functions and weights.

2.3.2. Offline Learning

A combination of primitives can be optimized offline without a need for the system model. For example, two primitives for the same goal point can be combined offline, where the weights, αs, used for the combination are added to 1, such that $\sum_{i=1}^{n} \alpha_i = 1$, where $\alpha > 0$. As shown in Section 2.1, a combined primitive is contracting, given that the original primitives are contracting. Using this setup, one can optimize the combination of the

weights in order to improve the performance of the resultant primitive. Subsequently, reinforcement learning can be applied to reshape the primitive again.

2.3.3. Maneuver Library

As defined in Appendix A, DMPs that we used are point attractors. In other words, the system converges to a goal state or equilibrium point. In real life, however, planning requires a change of maneuvers, and we require behaviors where transitions between primitives are needed. Using a controller that guarantees contraction, smooth transitions between primitives can be achieved. Therefore, a library of feasible trajectories must be stored, and we can exploit them in planning. We can create such library by learning feasible trajectories and generate more using the proposed methodology.

3. Application

3.1. The Combination of Primitives for a 3-DOF Helicopter

We used a simplified model derived from a Quanser Helicopter (see Figure 3) in our study. The helicopter is an under-actuated and minimum-phase system with two propellers at the end of its arm. Two DC motors are mounted below the propellers to create the forces which drive propellers. The motors' axes are parallel and their thrust is vertical to the propellers. There are three degrees of freedom: pitch (the vertical movement of the propellers), roll (the circular movement around the axis of the propellers), and travel (the movement around the vertical base) in contrast with conventional helicopters, which have six degrees of freedom.

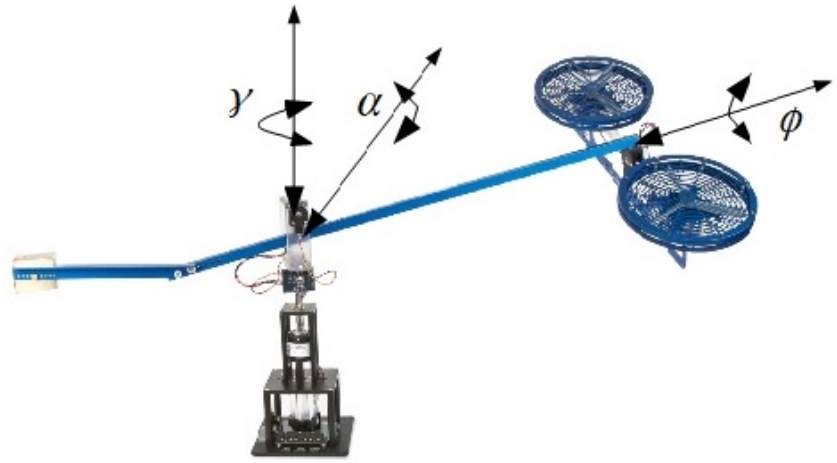

Figure 3. 3-DOF Quanser Helicopter [39].

In the model shown below (see details in [40]), the origin of the coordinate system is at the bearing and slip-ring assembly. The combinations of actuators form the collective ($T_{col} = T_L + T_R$) and cyclic ($T_{cyc} = T_L - T_R$) forces, and they are used as inputs in the system. The pitch and roll motions are controlled by collective and cyclic thrust, respectively. The motion in the travel angle is controlled by the components of thrust. A positive roll results in a positive change of angle.

$$\ddot{\alpha} = \frac{-Mgl_\alpha}{J_{yy}} \sin(\alpha + \alpha_0) + \frac{L}{J_{yy}} \cos(\phi) T_{col} \tag{29}$$

$$\ddot{\gamma} = \frac{L}{J_{zz}} \cos(\alpha) \sin(\phi) T_{col} - \frac{l_h}{J_{zz}} (T_L - T_R) \sin(\alpha) \sin(\phi) \tag{30}$$

$$\ddot{\phi} = \frac{l_h}{J_{xx}} T_{cyc} - \frac{mgl_\phi}{J_{xx}} \sin(\phi) \tag{31}$$

For such a system, feedback linearization (see details in [41]) is used. Control inputs are:

$$T_{cyc} = \frac{V_\phi - c_3 \sin(\phi)}{c_2} \tag{32}$$

$$T_{col} = \frac{V_\alpha - c_0 \sin(\alpha + \alpha_0)}{c_1 \cos(\alpha)}, \tag{33}$$

where V is the equivalent input, which can be calculated such that $V_\alpha = \ddot{\alpha}_d - 2\lambda_\alpha(\dot{\alpha} - \dot{\alpha}_d) - \lambda_\alpha^2(\alpha - \alpha_d)$. For the system model described above, aggressive maneuvers mimicking human-operator-generated maneuvers are designed to elevate the pitch angle, α, from -5.7° to 13.7° and 20.3° for a possible obstacle-avoidance problem. To reach a goal point between primitives, as shown in in Figure 2, primitives are combined in parallel using proper choices of α. Such a maneuver is followed by a 3-DOF Helicopter (see Figure 4). Here, the reference pitch angle reaches 16.7°.

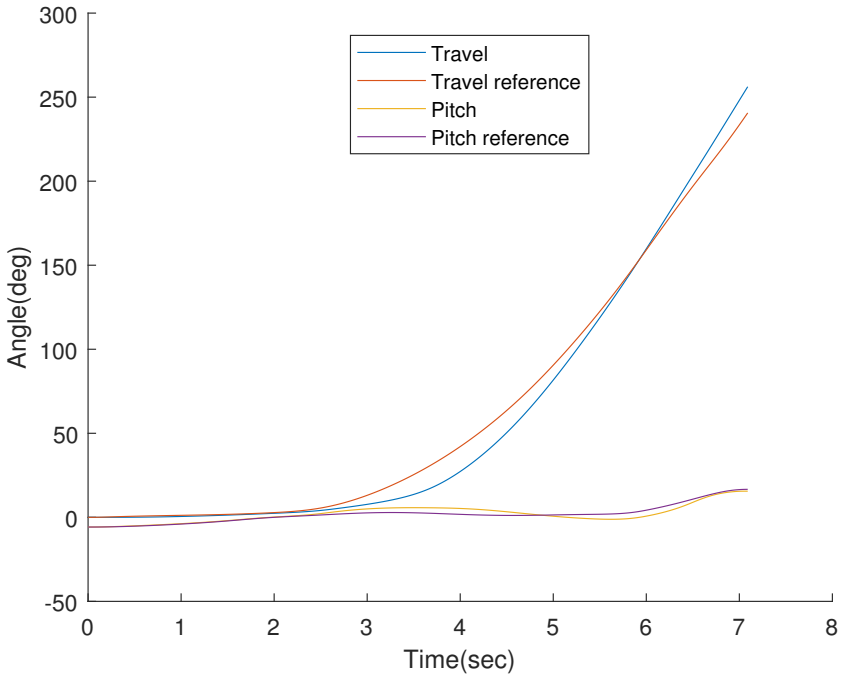

Figure 4. Tracking of pitch (α) and travel (γ) angles of a combined primitive.

3.2. MPC Application of Contraction Theory for DMPs

In application, a simple kinematics model will be used to characterize the planar motion of a UAV (see Figure 5). Although there are no dynamics involved, the model is still available to demonstrate nonholonomic motion. Model equations are shown below:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} \cos(\beta(t)) & 0 \\ \sin(\beta(t)) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ w(t) \end{bmatrix} \tag{34}$$

In this 2D model, x and y define the location of the UAV, and β is the heading angle. The controller inputs v and w represent the air and angular speed of the UAV, respectively.

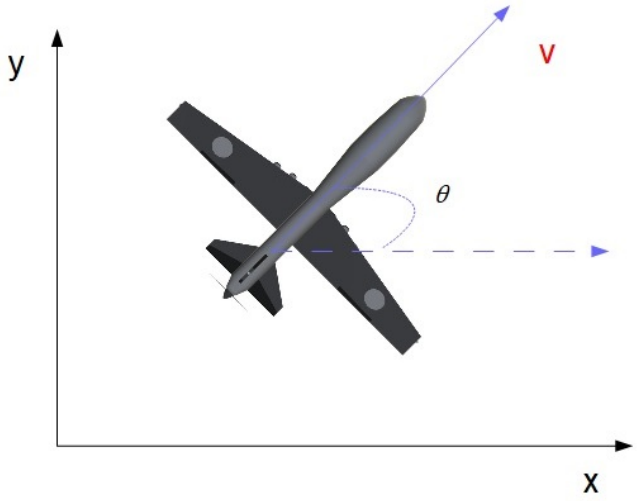

Figure 5. 2D Kinematics Model.

The problem can be reformulated as:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\beta} \end{bmatrix} = g_1(q)v + g_2(q)w = \begin{bmatrix} \cos(\beta(t)) \\ \sin(\beta(t)) \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} w. \tag{35}$$

The linearized system of the above kinematics cannot be controlled locally since the information of the nonlinear model is lost in system linearization. However, for the nonlinear system, the following result implies that the system is controllable:

$$rank[g_1\ g_2\ [g_1\ g_2]] = 3, \tag{36}$$

where $[g_1\ g_2]$ is the Lie bracket of vectors g_1 and g_2.

For the model, the desired commands are:

$$v_d(t) = \pm\sqrt{\dot{x}_d^2(t) + \dot{y}_d^2(t)} \tag{37}$$

$$w_d(t) = \frac{\ddot{y}_d(t)\dot{x}_d(t) - \ddot{x}_d(t)\dot{y}_d(t)}{\dot{x}_d^2(t) + \dot{y}_d^2(t)}, \tag{38}$$

where $w_d(t)$ is the differentiation of $\beta = atan2(\dot{y}, \dot{x})$. Using DMPs, the trajectories can be extracted for each state, and controllers can be generated thereafter.

In practice, we will derive feasible trajectories by combining contracting primitives (i.e., DMPs). Our system in Equation (34) can be written with desired inputs:

$$\dot{x} = v_d \cos(\beta) \tag{39}$$

$$\dot{y} = v_d \sin(\beta) \tag{40}$$

$$\dot{\beta} = w_d. \tag{41}$$

In this setup, β is contracting as w_d is contracting. Additionally, x and y are contracting, since the components of the system equations are contracting. Moreover, DMPs are extracted from feasible trajectories. Hence, we assume that the system is contracting in the close vicinity of the derived trajectories. These primitives can be combined such that:

$$\dot{x} = \alpha_1 B(x) u_{d_1} + \cdots + \alpha_n B(x) u_{d_n} = B(x)(\sum_{i=1}^{n} \alpha_i u_{d_i}), \tag{42}$$

where $\sum_{i=1}^{n} \alpha_i = 1$ and $\alpha_i > 0$. We can infer from Equations (37) and (38) that any parallel-combined primitive will be within the controller limits of the system. From construction of DMP formulas, we can find that the equilibrium point will be $\sum_{i=1}^{n} \alpha_i g_i$ for DMPs.

For such a system, it is also possible to use feedback linearization. For input–output linearization [42], we define an output vector, $\eta = (x, y)$, and differentiate the output vector two times in order to find input, thus:

$$\ddot{\eta} = \dot{\xi} \begin{bmatrix} \cos(\beta) & -\xi \sin(\beta) \\ \sin(\beta) & \xi \cos(\beta) \end{bmatrix} \begin{bmatrix} a \\ w \end{bmatrix}, \tag{43}$$

where an integrator with a state $\xi = v$ is added. Defining the system as:

$$\begin{bmatrix} a \\ w \end{bmatrix} = \begin{bmatrix} \cos(\beta) & -\xi \sin(\beta) \\ \sin(\beta) & \xi \cos(\beta) \end{bmatrix}^{-1} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \tag{44}$$

one should note that the resulting controller has a singularity point where $\xi = v = 0$.

$$u_1 = \ddot{x}_d(t) + k_{p1}(x_d(t) - x) + k_{d1}(\dot{x}_d(t) - \dot{x}) \tag{45}$$

$$u_2 = \ddot{y}_d(t) + k_{p2}(y_d(t) - y) + k_{d2}(\dot{y}_d(t) - \dot{y}) \tag{46}$$

Through the construction of the DMP formulas, the system will converge to a weighted desired location, $\sum_{i=1}^{n} \alpha_i g_i$, if the desired end points, $(g_i s)$, are different.

In practice, the flight maneuver defined for (x, y, β) starts from the origin, $(0, 0, 0)$, and ends at specified goal point $(1.5, 1.5, 0)$. Firstly, the MPC application [43] using the CasADi [44] optimization tool is used to generate trajectories (Figure 6).

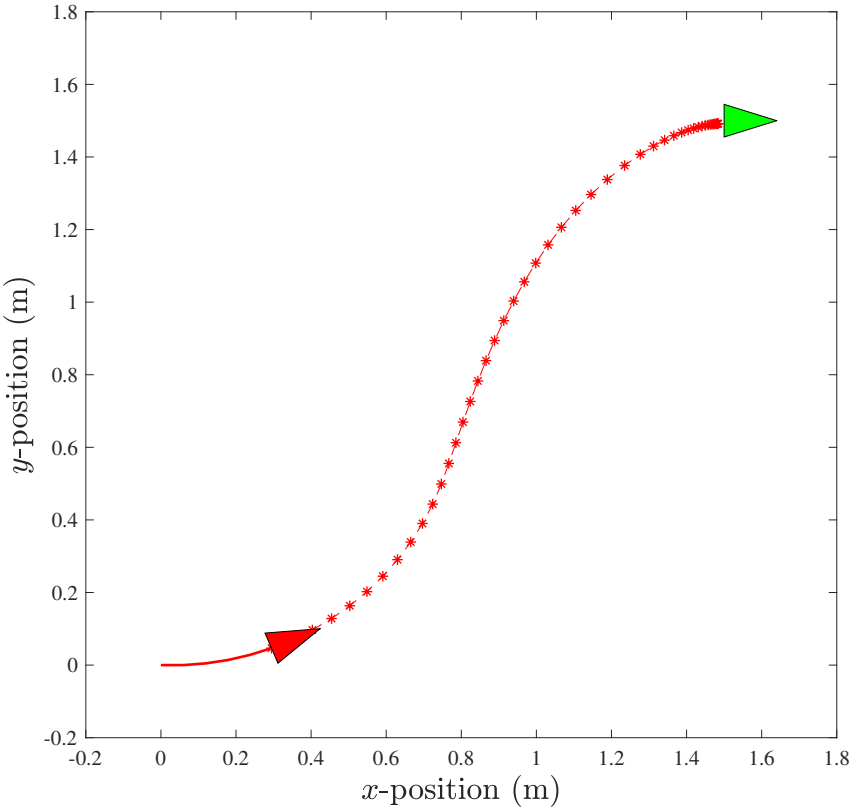

Figure 6. Flight maneuver where * and ▷ denote prediction horizon and UAV (red: current posture; green: desired posture), respectively.

Then, the DMP is used to imitate each primitive (see Figure 7).

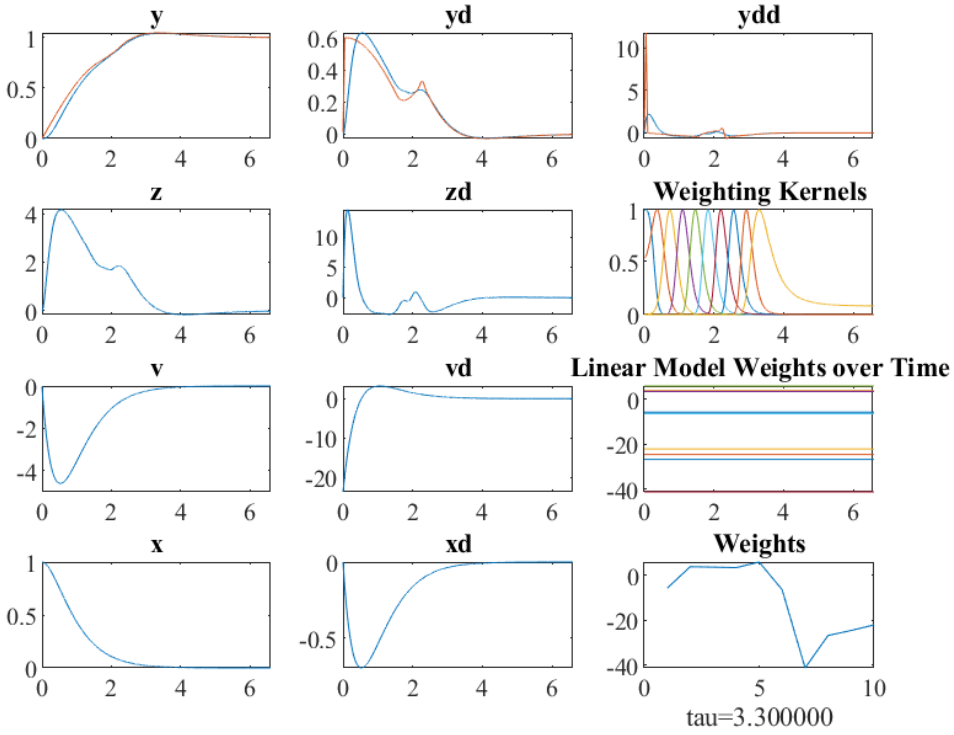

Figure 7. DMP internals for x-axis.

For parallel combinations, two primitives, as shown in Figure 8, with goal points $(1, 1, 0)$ and $(1.5, 1.5, 0)$ are combined using a weighting parameter, α_i, to generate a primitive with a goal point $(1.25, 1.25, 0)$.

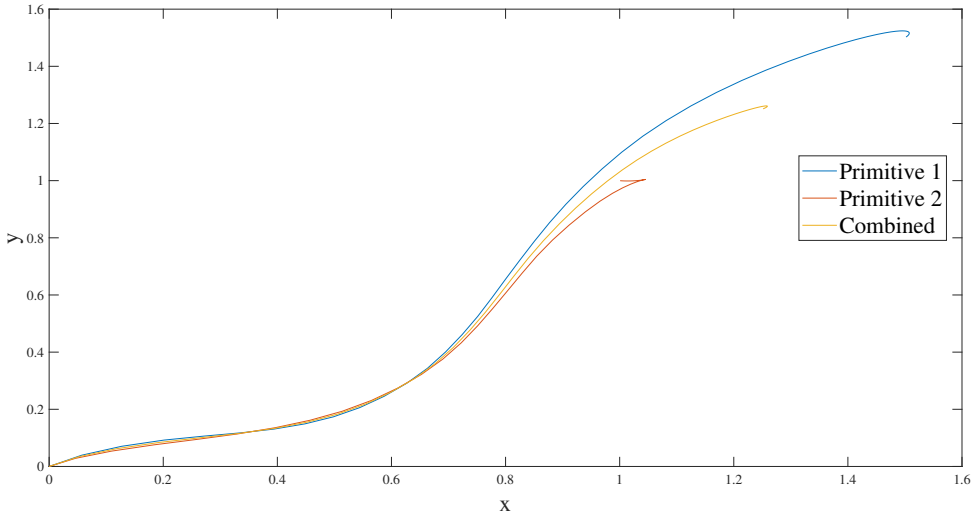

Figure 8. Combination of DMPs.

Finally, MPC is once again used to follow the combined feasible trajectory (See Figure 9).

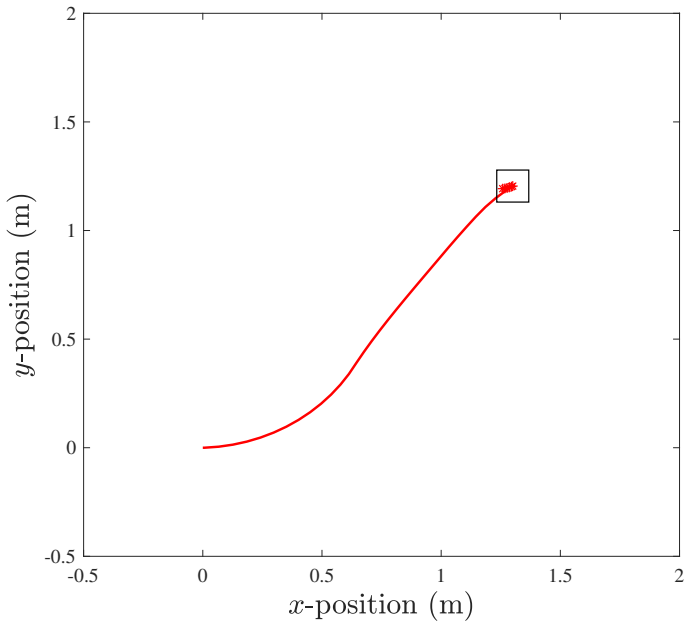

Figure 9. MPC to follow a new trajectory where □ denotes the UAV.

3.3. The Application of Contraction Theory in Reinforcement Learning (PI^2)

For application, the PI^2 reinforcement learning software [45] is used for a system with a kinematics model defined in Equation (34). For an initial primitive of a flight maneuver with an end point $(1.5, 1.5, 0)$, the DMP imitated the trajectories generated by the MPC algorithm [43]. A total of 100 updates and 10 basis functions/weights was applied. For each update, 10 trials were used. We applied importance sampling in which the five best trials from the previous update were rerun in the next update. The cost function is:

$$J = \frac{1}{2}\dot{\mathbf{x}}^{\mathbf{T}}\dot{\mathbf{x}}Q + \frac{1}{2}\mathbf{w}^{\mathbf{T}}\mathbf{w}R + r_T, \tag{47}$$

where $\mathbf{x}$ and $\mathbf{w}$ are the velocity state vector and weights, respectively. The constants $Q = 1000$, and $R = 1$ are used for penalization. The terminal cost is shown as r_T. The resulting trajectory for x and y is shown in Figure 10.

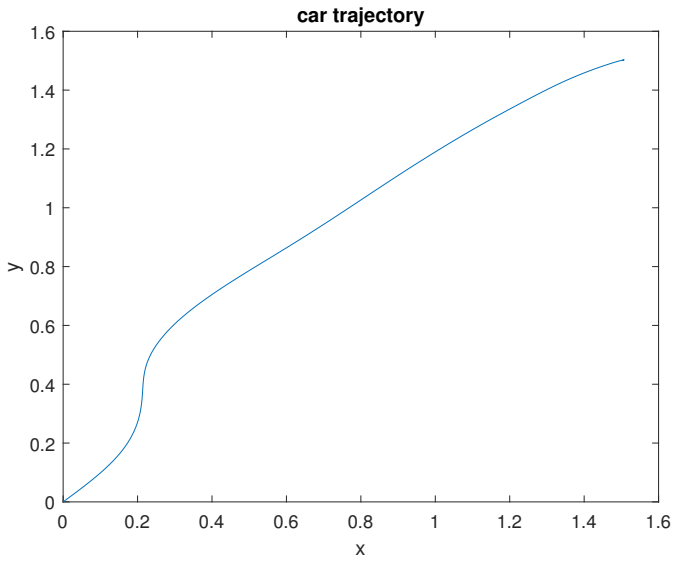

Figure 10. Flight maneuver where goal point is $(1.5, 1.5)$.

We can also imitate and learn a primitive for a goal point, $(1, 1)$, using same process. Learning curves for both primitives are shown in Figure 11. After training for both primitives, the primitives were then combined in parallel as defined in Equation (25) in order to generate a stable trajectory, which has a goal point $(1.25, 1.25)$. We observe that learning is transferred in the combined primitive (see Figure 12).

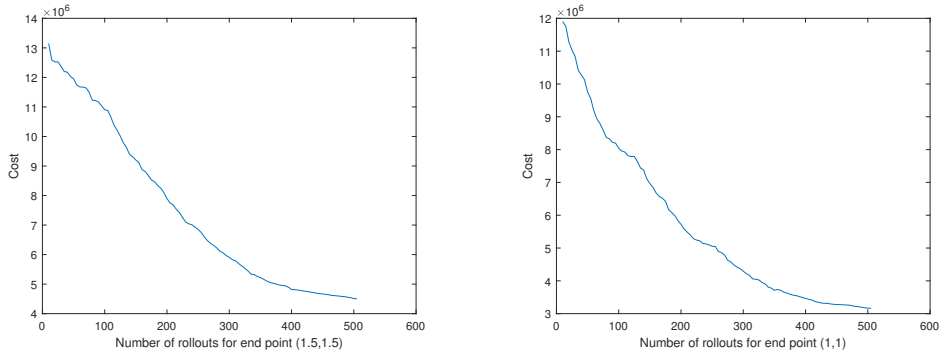

Figure 11. Learning curves for flight maneuver primitives.

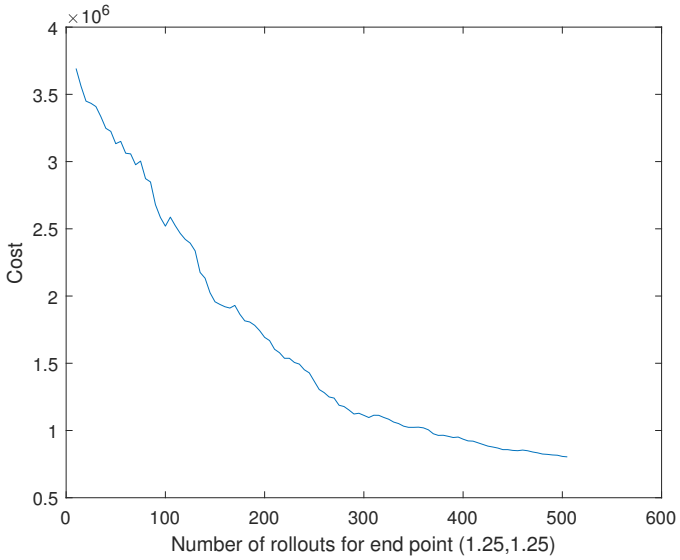

Figure 12. Learning curve for a combined primitive.

4. Conclusions

Our aim in this paper was to propose a framework that combined several methods to generate maneuvers inspired by biological experiments. In our methodology, stability is addressed with contraction theory, and reinforcement learning is used to improve maneuvers in contraction regions. Our main results are demonstrated on 3-DOF-helicopter and 2D-UAV models. We believe that learning in contraction regions is a key aspect for achieving stable trajectories. Our approach exploits such contracting regions using a combination of primitives. It is shown that trajectories updated by reinforcement learning are feasible, and learning is transferred in contraction regions. It is also possible to employ several other methods, as discussed in our paper, to study the contraction regions around trajectories.

As a part of our future research, we aim to apply this methodology to actual flight platforms. We will focus particularly on scenario-based problems, such as obstacle avoidance, in which serial and parallel combinations of different primitives will be deployed using a maneuver library.

Author Contributions: Conceptualization: B.E.P. and G.I.; methodology: B.E.P. and G.I.; software, B.E.P.; validation: B.E.P. and G.I.; investigation: B.E.P. and G.I.; resources: B.E.P. and G.I.; writing–original draft preparation: B.E.P.; writing–review and editing: B.E.P. and G.I.; visualization: B.E.P. and G.I.; supervision: G.I.; project administration: B.E.P. and G.I. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

In DMPs [34], the goal is to reach an attractor state. A trajectory is generated where transient states of the system are learned by the combination of weights and basis functions. The DMP for a discrete movement can be summarized as follows:

$$\tau \dot{z} = \alpha_z(\beta_z(g - y) - z) \tag{A1}$$

$$\tau \dot{y} = z + f, \tag{A2}$$

where y, $\dot{y}$, and $\ddot{y}$ represent the desired trajectory; α_z and β_z are time constants; τ is a temporal scaling factor; and g is the desired goal state. Additionally, a second-order dynamic system can be introduced:

$$\tau \dot{v} = \alpha(\beta_z(g - x) - v) \tag{A3}$$

$$\tau \dot{x} = v. \tag{A4}$$

From this setup, it can be shown that system will converge to an attractor point g, when the f-function is assumed to be 0. In DMP, a nonlinear function f is modified in order to learn the desired trajectories between the start and end points.

In Equation (A2), f is a linear combination Gaussian weighting kernels such that:

$$f(x, v, g) = \frac{\sum_{i=1}^{N} \Psi_i w_i v}{\sum_{i=1}^{N} \Psi_i}, \tag{A5}$$

where:

$$\Psi_i = exp\{-h_i(x/g - c_i)^2\}, \tag{A6}$$

and h_i, c_i, and w_i represent the bandwidth, the center of the Gaussian kernels, and the weights, respectively.

Appendix B

In general, the stability of the feasible trajectories can be studied using several methods summarized below.

Appendix B.1. Basic Contraction Properties and Matrix Measures

In basic contraction theory, the Jacobian, $\frac{\partial f}{\partial x}$, should be a uniformly negative definite for system to contract. In other words:

$$\exists \beta > 0, \forall \mathbf{x}, \forall t \geq 0, \frac{1}{2}\left(\frac{\partial f}{\partial x} + \frac{\partial f^T}{\partial x}\right) \leq \beta \mathbf{I} < 0. \tag{A7}$$

In [46], this result is extended by matrix measures [47]:

$$\mu(A) = \lim_{x \to 0^+} \frac{\|I + tA\| - \|I\|}{t}, \tag{A8}$$

where Equation (A7) corresponds to $\mu_2(A)$. Matrix measures were used to study the reachability analysis. Furthermore, the Toolbox for Interval Reachability Analysis (TIRA) [48], a MATLAB library that gathers several methods for reachability analysis, is proposed. Following the contraction/growth bound method in TIRA, a state vector is partitioned into components to study reachability.

Appendix B.2. Generalized Contraction Analysis Using Metrics

$\delta z = \Theta \delta x$ is defined to derive generalized contraction theory [30]. For a system such as those represeented in Equations (4) and (5), a dual metric $W(x) = M(x)^{-1} = (\Theta^T \Theta)^{-1}$ is calculated using the sum of squares [49], and differential control is constructed as $\delta u = -K(x)\delta x$, with $K = \frac{1}{2}\rho B' W^{-1}$ [50].

Appendix B.3. Global Metrics Derived Using Linearization

At equilibrium, one can linearize the system in the form of $\dot{x} = f(x)$, and the resulting equation can be defined as $\dot{x} = A(x,t)x$. As mentioned in [30], a coordinate transformation, $z = \Theta x$, (Θ is constant) can be formulated for a Jordan form, thus:

$$\dot{z} = \Theta A \Theta^{-1} z = \Lambda z. \tag{A9}$$

As a result, a generalized Jacobian becomes $F = \Lambda$. This is due to the fact that the system in the equilibrium point, A, should have negative eigenvalues. Therefore, the existence of θ is guaranteed, and one can apply this metric to find the region of contraction.

Appendix B.4. Combination of Primitives

For the contracting systems defined below:

$$\dot{x}_1 = f_1(x_1, t), \ldots \dot{x}_n = f_n(x_n, t), \tag{A10}$$

one can combine their virtual dynamics, as proposed in [30], with positive αs such that:

$$\alpha_1(t)\dot{\delta}x_1 + \ldots + \alpha_n(t)\dot{\delta}x_n, \tag{A11}$$

and the combined system will also contract. It is possible to use this methodology to combine primitives. For an identical system, different primitives are defined below:

$$\begin{aligned} \dot{x}_1 &= f(x_1, t) + K(x_1 - x_{ref_1}) \\ &\vdots \\ \dot{x}_n &= f(x_n, t) + K(x_n - x_{ref_n}). \end{aligned}$$

If the system is combined such that αs are positive and $\sum_{i=1}^{n} \alpha_i = 1$, then the particular solutions of each system will contract to the reference trajectory. In other words, the parallel combined system will contract to the linear combination of a reference system, since $\sum_{i=1}^{n} \alpha_i f(x) = f(x)$ represents the same system.

References

1. Yuksek, B.; Demirezen, U.; Inalhan, G.; Tsourdos, A. Cooperative Planning for an Unmanned Combat Aerial Vehicle Fleet Using Reinforcement Learning. *J. Aerosp. Inf. Syst.* **2021**, *18*, 739–750. [CrossRef]
2. Herekoglu, O.; Hasanzade, M.; Saldiran, E.; Cetin, A.; Ozgur, I.; Kucukoglu, A.; Ustun, M.; Yuksek, B.; Yeniceri, R.; Koyuncu, E.; et al. Flight Testing of a Multiple UAV RF Emission and Vision Based Target Localization Method. In Proceedings of the AIAA Scitech 2019 Forum, San Diego, CA, USA, 7–11 January 2019. [CrossRef]

3. Karali, H.; İnalhan, G.; Demirezen, M.; Yükselen, M. A new nonlinear lifting line method for aerodynamic analysis and deep learning modeling of small unmanned aerial vehicles. *Int. J. Micro Air Veh.* **2021**, *13*. [CrossRef]
4. Gavrilets, V.; Frazzoli, E.; Mettler, B.; Piedmonte, M.; Feron, E. Aggressive Maneuvering of Small Autonomous Helicopters: A Human Centered Approach. *Int. J. Robot.* **2001**, *20*, 795–807. [CrossRef]
5. Coates, A.; Abbeel, P.; Ng, A. Learning for Control from Multiple Demonstrations. In Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, 5–9 July 2008; pp. 144–151.
6. Lupashin, S.; Schöllig, A.; Sherback, M.; D'Andrea, R. A simple learning strategy for high-speed quadrocopter multi-flips. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AL, USA, 3–8 May 2010; pp. 1642–1648.
7. Levin, J.; Paranjabe, A.; Nahon, M. Agile maneuvering with a small fixed-wing unmanned aerial vehicle. *Robot. Auton. Syst.* **2019**, *116*, 148–161. [CrossRef]
8. Guerrero-Sánchez, M.; Hernández-González, O.; Valencia-Palomo, G.; López-Estrada, F.; Rodríguez-Mata, A.; Garrido, J. Filtered Observer-Based IDA-PBC Control for Trajectory Tracking of a Quadrotor. *IEEE Access* **2021**, *9*, 114821–114835. [CrossRef]
9. Xiao, J. Trajectory planning of quadrotor using sliding mode control with extended state observer. *Meas. Control* **2020**, *53*, 1300–1308. [CrossRef]
10. Almakhles, D. Robust Backstepping Sliding Mode Control for a Quadrotor Trajectory Tracking Application. *IEEE Access* **2020**, *8*, 5515–5525. [CrossRef]
11. Yuksek, B.; Inalhan, G. Reinforcement learning based closed-loop reference model adaptive flight control system design. *Int. J. Adapt. Control. Signal Process.* **2021**, *35*, 420–440. [CrossRef]
12. Phung, M.; Ha, Q. Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization. *Appl. Soft Comput.* **2021**, *107*, 107376. [CrossRef]
13. Li, Y.; Liu, C. Efficient and Safe Motion Planning for Quadrotors Based on Unconstrained Quadratic Programming. *Robotica* **2021**, *39*, 317–333. [CrossRef]
14. Lee, K.; Choi, D.; Kim, D., Potential Fields-Aided Motion Planning for Quadcopters in Three-Dimensional Dynamic Environments. In Proceedings of the AIAA Scitech 2021 Forum, Nashville, TN, USA, 11–15 January 2021.
15. Zhang, X.; Shen, H.; Xie, G.; Lu, H.; Tian, B. Decentralized motion planning for multi quadrotor with obstacle and collision avoidance. *Int. J. Intell. Robot. Appl.* **2021**, *5*, 176–185. [CrossRef]
16. Chow, Y.; Nachum, O.; Duenez-Guzman, E. A Lyapunov-based approachto safe reinforcement learning. In Proceedings of the NIPS 2018, Montréal, QC, Canada, 3–8 December 2018; pp. 8103–8112.
17. Wenqi, C.; Zhang, B. Lyapunov-regularized reinforcement learning for power system transient stability. *arXiv* **2021**, arXiv:2103.03869.
18. Perkins, T.; Barto, A. Lyapunov design for safe reinforcement learning. *J. Mach. Learn. Res.* **2002**, *3*, 803–832.
19. Jadbabaie, A.; Hauser, J. On the stability of receding horizon control with a general terminal cost. *IEEE Trans. Autom. Control* **2005**, *50*, 674–678. [CrossRef]
20. Mehrez, M.; Worthmann, K.; Mann, G.; Gosine, R.; Faulwasser, T. Predictive path following of mobile robots without terminal stabilizing constraints. In Proceedings of the 20th IFAC World Congress, Toulouse, France, 11–17 July 2017; pp. 10268–10273.
21. Grüne, L.; Pannek, J.; Seehafer, M.; Worthmann, K. Analysis of unconstrained nonlinear MPC schemes with varying control horizon. *SIAM J. Control. Optim.* **2010**, *48*, 4938–4962. [CrossRef]
22. Galef, B. Imititaion in Animals: History, Definition and Interpretation of Data from the Psychological Laboratory. In *Comparative Social Learning*; Psychology Press: Hove, UK, 1988; pp. 3–28.
23. Polit, A.; Bizzi, E. Characteristic of Motor Programs Underlying Arm Movements in Monkeys. *J. Neurophsiology* **1979**, *42*, 183–194. [CrossRef]
24. Bizzi, E.; Mussa-Ivaldi, F.; Hogan, N. Regulation of multi-joint arm posture and movement. *Prog. Brain Res.* **1986**, *64*, 345–351.
25. Mussa-Ivaldi, F.; Giszter, S.F.; Bizzi, E. Motor Space Coding in the Central Nervous System. *Cold Spring Harb. Symp. Quant. Biol.* **1990**, *55*, 827–835. [CrossRef]
26. Bizzi, E.; Mussa-Ivaldi, F.; Giszter, S. Computations underlying the execution of movement: A biological perpective. *Science* **1991**, *253*, 287–291. [CrossRef]
27. Mussa-Ivaldi, F.A.; Giszter, S.F.; Bizzi, E. Linear combinations of primitives in vertebrate motor control. *Proc. Natl. Acad. Sci. USA* **1994**, *91*, 7534–7538. [CrossRef]
28. Mussa-Ivaldi, F.A.; Bizzi, E. Motor learning through the Combination of Primitives. *Philos. Trans. R. Soc. B Biol. Sci.* **2000**, *355*, 1755–1769. [CrossRef] [PubMed]
29. Schaal, S.; Mohajerian, P.; Ijspeert, A. Dynamics systems vs. optimal control—A unifying view. *Prog. Brain Res.* **2007**, *165*, 425–445. [PubMed]
30. Lohmiller, J.S.W. On Contraction Analysis for Nonlinear Systems. *Automatica* **1998**, *34*, 683–686. [CrossRef]
31. Bazzi, S.; Ebert, J.; Hogan, N.; Sternad, D. Stability and predictability in human control of complex objects. *Chaos Interdiscip. J. Nonlinear Sci.* **2018**, *28*, 103103. [CrossRef]
32. Theodorou, E.; Buchli, J.; Schaal, S. A generalized path integral controlapproach to reinforcement learning. *J. Mach. Learn. Res.* **2010**, *11*, 3137–3181.
33. Perk, B.E.; Slotine, J.J.E. Motion primitives for robotic flight control. *arXiv* **2006**, arXiv:cs/0609140v2.

34. Ijspeert, A.; Nakanishi, J.; Schaal, S. Learning attractor landscapes for learning motor primitives. In *Advances in Neural Information Processing Systems 15*; MIT Press: Cambridge, MA, USA, 2003; pp. 1547–1554.
35. Slotine, J.; Li, W. *Applied Nonlinear Control*; Prentice-Hall: Hoboken, NJ, USA, 1991.
36. Suttojn, R.; Barto, A. *Reinforcement Learning*; MIT Press: Cambridge, MA, USA, 1998.
37. Stengel, R. *Optimal Control and Estimation*; Dover Books on Advanced Mathematics; Dover Publications: New York, NY, USA, 1994.
38. Fleming, W.; Soner, H. Controlled Markov Processes and Viscosity Solutions. In *Applications of Mathematics*, 2nd ed.; Springer: New York, NY, USA, 2006.
39. Available online: https://www.quanser.com (accessed 20 January 2022).
40. Ishutkina, M. Design and Implimentation of a Supervisory Safety Controller for a 3DOF Helicopter. Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2004.
41. Perk, B.E. Control Primitives for Fast Helicopter Maneuvers. Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2006.
42. Luca, A.; Oriola, G.; Vendittelli, M. Control of wheeled mobile robots: An experimental overview. In *Ramsete*; Nicosia, S., Sicil, B., Bicchi, A., Valigi, P., Eds.; Springer: Berlin, Germany, 2001; Volume 270, pp. 181–226.
43. Mehrez, M. Github. MPC and MHE Implementation in MATLAB Using Casadi. Available online: https://github.com/MMehrez (accessed 20 January 2022).
44. Andersson, J.; Gillis, J.; Horn, G.; Rawlings, J.; Dieh, M. CasADi: A Software Framework for Nonlinear Optimization and Optimal Control. *Math. Program. Comput.* **2019**, *11*, 1–36. [CrossRef]
45. Theodorou, E.; Buchli, J.; Schaal, S. Path Integral Reinforcement (PI^2) Learning Software. Available online: http://www-clmc.usc.edu/Resources/Software (accessed 20 January 2022).
46. Maidens, J.; Arcak, M. Reachability analysis of nonlinear systems using matrix measures. *IEEE Trans. Automat. Control* **2015**, *60*, 265–270. [CrossRef]
47. Desoer, C.; Vidyasagar, M. *Feedback Systems: Input-Output Properties*; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2009.
48. Meyer, P.; Davenport, A.; Arcak, M. TIRA: Toolbox for interval reachability analysis. *arXiv* **2019**, arXiv:1902.05204.
49. Aylward, E.; Parrilo, P.; Slotine, J. Stability and robustness analysis of nonlinear systems via contraction metrics and SOS programming. *Automatica* **2008**, *48*, 2163–2170. [CrossRef]
50. Manchester, I.; Tang, J.; Slotine, J. Unifying robot trajectory tracking with control contraction metrics. In *Robotics Research*; Springer: Cham, Switzerland, 2018; pp. 403–418.