Hybrid Modeling and Control for an Autonomous Passenger Car

André Miguel Guerreiro Carvalho
andre.miguel.c@tecnico.ulisboa.pt
Instituto Superior Técnico
Lisbon, Portugal

***Abstract*— The popularity of autonomous vehicles has been gaining momentum in the last few years. Safety assessment of such systems is of utmost importance for a commercial certification. In this work, we will be modeling the vehicle as an Hybrid System - combining discrete states where in one of them, the system displays a different continuous-time dynamics. This thesis presents an architecture to control such system, inspired on how humans drive, and, intuitively embed uncertainty. We employ ideas from Viability Theory, such as viability kernels, to restrict the system's inputs to the safe ones. The viability kernel is the largest subset of the constraint set where at least one trajectory starting there remains inside the constraint set for some time. We improved an existing algorithm to compute finite-time viability kernels, based on sampling the boundary of the constraint set. The novel feature implies that the algorithm is faster than the previous ones and is naturally extended to non-linear systems described by a first order, continuous, differential equation, convex in the input variables. Proofs validating the properties of the novel algorithm are presented. We claim that, for non-linear systems and in some conditions, our algorithm has better asymptotic complexity than the usual griding methods. Results on the convexity of the infinite-time viability kernel for some class of differential inclusions are also given.**

I. Introduction

A. Motivation

The topic of autonomous mobile robots is a well covered subject in control and robotics. Their purposes range from surveillance of coastal zones [1], warehouse management systems [2] and, the one approached in this work, autonomous passenger cars[3].

The popularity of this field, among general audience has raised dramatically since 2012, (see figure 1 below), making this a trendy topic since then.

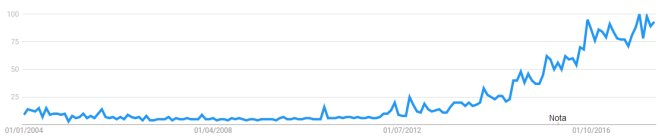

Fig. 1. Search popularity for the topic "Autonomous Vehicles" in Google search engine. 100 being the relative peak in popularity.

Autonomous vehicles are already well established in our day to day life. Trains and subways are mostly autonomous, with its driver having smaller and smaller influence[4], in an airplane flight, most of the flight time is done in autopilot mode [5].

Fully autonomous road vehicles could be the next major breakthrough in urban transportation. Buses may be more often on schedule, delays due to traffic are mitigated since time of arrivals can be more accurately predicted. Fuel efficiency and increasing road capacity are also two big advantages of these vehicles[6]. The possibility of reducing car crashes trough elimination of human errors[7], since they cause 90% of car crashes [8], is also a major driver in the development of this technology.

Safety assessment of such self-driving vehicles becomes of extreme importance for bigger acceptance of such systems. Although most people feel moderately safe with autonomous vehicles [9], the guarantee that the vehicle will avoid most accidents - and not making them happen when a human driver would not - plays a major role in the spreading of usage of these cars. In this work we present an alternative approach to the safety assessment and propose a control architecture for a self-driving car.

B. Contributions

This thesis describes an autonomous car more closely to what a competent human driver would do. This description consists in looking at the problem of driving a car in a more broader sense - we are interested in a sequence of sets of actions possible to safely drive the car instead of a single sequence of inputs. Human drivers typically have a mental model of a set of positions of the steering wheel and of the accelerator pedal that are somewhat safe for their level of skill or willingness to risk [10]. This mental process is what will inspire the architecture for the intelligent part of the vehicle and is related to the scope of *Viability Theory*.

The main focus of the thesis is on the safety assessment of the controls employed in the vehicle recurring to *Viability Theory* [11]. This theory is based on the notion of viable solutions of the differential inclusion describing the system - solutions that stay inside a safe constraint set for a finite or infinite time horizon. The hybridization of the system comes from the fact that at each sampling time we are restricting the set of viable inputs, therefore, the differential inclusion changes at each instant. Our interest will be mainly on systems described by differential inclusions that are guaranteed to have solutions in convex constraint sets.

The largest subset contained in the constraint set in which at least one trajectory starting there remains there for some time is called the *viability kernel*. We've adapted existing algorithms to approximate viability kernels for linear systems,

making them faster for a high number of dimensions (up to 100 dimensions), and extendable to non-linear systems described by first-order, continuously differentiable, differential equations. The algorithm is based on sampling points in the boundary of the constraint set and assessing the points closest to that boundary that belong to the viability kernel by forward simulation. We proved the algorithm works with our adaptation in the linear case and established the conditions for it to work in the non-linear cases. Proofs on the convexity of the viability kernel are also presented.

II. Proposed Architecture

Taking into account how a driver behaves and how the flow of information must occur in a autonomous vehicle, we propose a top-level architecture for the intelligent part of the vehicle (Figure 2). This model encompasses the main idea of how the human brain works while performing these kind of tasks. Since this thesis will be focused essentially on the *Local Planning* module, only that module will have a detailed explanation - such as how to approximate the viability kernels - in section IV.

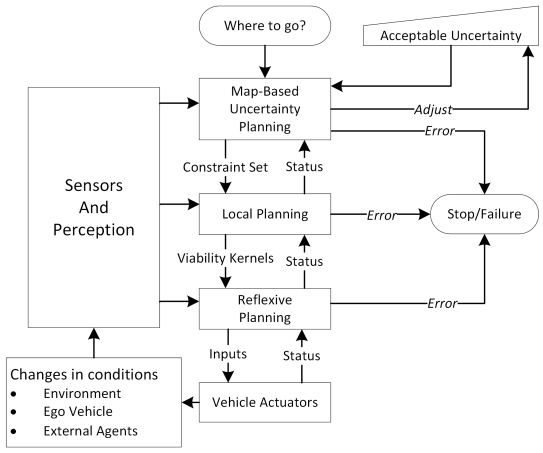

Fig. 2. Top-Level Architecture of the cognitive part of the vehicle

The system starts by determining the vehicle's state, where the passenger would like to go and its acceptable uncertainty. It acts upon a tuning knob, adjusted by the user, selecting the acceptable uncertainty for the trip. Every system has some probability of occurring an error. Faulty data and too much loose tolerances are examples of things responsible for making the vehicle misbehave. Increasing the acceptable uncertainty 'tells' the system to lower the accuracy of computations and thus making the input set 'bigger', allowing higher velocities. It can happen that the system, for some route, hits its lower bound of uncertainty and enforces to be chosen a 'safer' uncertainty. This feedback may occur at any stage and if the system is already performing the mission, it should alert the passenger and initiate a safe stop procedure.

A. Local Planning

At this layer, the exact set of inputs allowed for safely driving the car, based on the viability kernel, is computed. Such inputs are defined as a feedback map,

$$R(x_k) := \{u_k \in \mathcal{U} \mid x_{k+1} \in \text{Viab}(K)\}. \tag{1}$$

This means that, for some state x_k, one will reduce the original input set, $\mathcal{U}$ to $R(x_k)$. This reduced set is the set of all inputs that allow the state of the system to be inside the viability kernel, Viab(K) in the next sampling instant, x_{k+1}. The computations in this layer can and should be done before the vehicle starts its mission, since these are complex and will eventually take some time. An illustration of this process of selecting viable trajectories can be seen in figure 3.

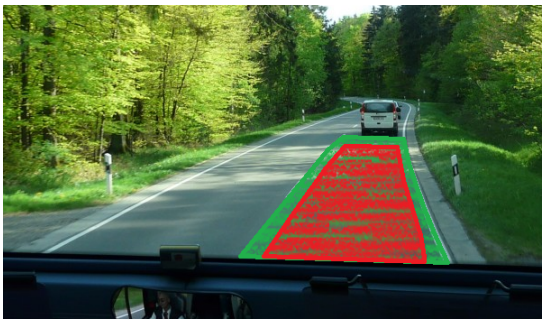

Fig. 3. Illustration of the viability kernel in red, and the outermost set in green, the constraint set

One aspect of having an architecture lied out like this is that if there are only a finite number of uncertainty levels for the user to choose, all this information up to this point, *Map-Based Planning* and *Local Planning*, can be stored in memory and downloaded after the previous layer decided the route the vehicle should take. Online change of routes is still possible, since it suffices to extract this information from the newly included map segments.

The hybrid aspect of the vehicle comes from the fact that at each sampling time the dynamics of the vehicle change due to a change in the input space. The differential inclusion that represents the vehicle at each time is modified in order to keep the vehicle inside the viability kernel. The respective set-valued map, after an adequate discretization of the system[12], becomes,

$$F_k(x_k) := \{f(x_k, u_k)\}_{u_k \in R(x_k)}. \tag{2}$$

Which leads to the difference inclusion modeling the system,

$$x_{k+1} \in F_k(x_k). \tag{3}$$

Non-emptiness of $R(x_k)$ has to be guaranteed by the program itself, meaning, as soon as the set of usable inputs is the empty set the program sends a failure signal to the layer below and starts a safe stopping procedure. The user is then advised to choose a more tolerant uncertainty level.

III. BACKGROUND

The systems considered in the paper are of the form,

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)), & \text{for almost all } t \\ u(t) \in \mathcal{U}, \quad x(0) = x_0 \in \mathcal{X} \end{cases} \tag{4}$$

Where, $\mathcal{X}$,$\mathcal{U}$ are sets in a metric space standing, respectively, for a state space and a compact control space. Both are considered non-empty.

The conditions for existence and uniqueness of the solutions of the differential inclusion (4) are discussed below. Consider a set-valued map $F : \mathcal{X} \rightsquigarrow \mathcal{Y}$, with $\mathcal{Y}$ a set in a metric space, defined as,

$$\forall x \in \mathcal{X}, \quad F(x) := \{f(x,u)\}_{u \in \mathcal{U}} = \bigcup_{u \in \mathcal{U}} f(x,u). \tag{5}$$

This is a common extension of a wide class of differential equation models accounting for uncertainties (see for instance [13], pp. *xiv*). We can reformulate the system (4) as,

$$\dot{x}(t) \in F(x(t)), \qquad \text{for almost all } t > 0. \tag{6}$$

Continuity properties play a fundamental role in the existence of solutions of systems in the form (4). These are briefly reviewed below.

Definition 1. ([14]) F is *upper semicontinuous* (USC) at $\bar{x} \in X$ if for any open N containing $F(\bar{x})$ there exists a neighborhood M of $\bar{x}$ such that $F(M) \subset N$. F is a upper semicontinous set-valued map if it is USC for every $\bar{x} \in X$.

Definition 2. ([14]) F is *lower semicontinuous* (LSC) at $\bar{x} \in \mathcal{X}$ if for any $\bar{y} \in F(\bar{x})$ and any neighborhood $N(\bar{y})$, there exists a neighborhood $M(\bar{x})$ of $\bar{x}$ such that

$$\forall x \in M(\bar{x}), \qquad F(x) \cap N(\bar{y}) \neq \emptyset \tag{7}$$

F is a lower semi-continuous set-valued map if it is LSC for every $\bar{x} \in X$

The following result summarizes the continuity properties of the dynamics map that are key to the existence of solutions of (6).

Theorem 1. *Consider (5), with $\mathcal{X}$, $\mathcal{Y}$, $\mathcal{U}$ subsets of metric spaces. If (i) $f : \mathcal{X} \times \mathcal{U} \mapsto \mathcal{Y}$ is a Lipschitz function in both variables, (ii) for each $x \in \mathcal{X}$, f is convex relative to the second variable, u, and (iii) $\mathcal{U}$ is a non-empty, compact, convex set, then $\forall x \in \mathcal{X}, F(x)$ is a continuous (LSC and USC) and Lipschitz set-valued map, with non-empty, compact and convex images.*

Proof: The demonstration follows by showing, in sequence, non-emptiness, compactness, convexity, Lipschitziness[1] and each semi-continuity.

Non-emptiness:

Since $\mathcal{U} \neq \emptyset$ and $f(\cdot)$ is a continuous function in both variables, the set $F(x)$ has always at least one element, thus, it is non-empty.

Compactness:

The subset $\mathcal{U}$ being compact means that every sequence, u_n, in $\mathcal{U}$ has a convergent subsequence. By definition of uniform continuity, (see for instance [15]), with n a sucession index,

$$\forall (x, u_n)_{n \in \mathbb{N}} \subset \mathcal{X} \times \mathcal{U}, \tag{8}$$

$$\lim_{n \to \infty} (x, u_n) = (x, \bar{u}) \implies \lim_{n \to \infty} f(x, u_n) = f(x, \bar{u}). \tag{9}$$

This means that, for every convergent subsequence in $\mathcal{U}$, we have a corresponding convergent subsequence $f(x, u_n) \in F(x)$. Therefore, every sequence of elements in $F(x)$ has at least one convergent subsequence, the set is then sequentially compact (by definition of compactness). Every sequentially compact metric space is compact [2], so, $F(x)$ is compact $\forall x \in \mathcal{X}$.

Convexity:

The convexity follows from the fact that $\forall x \in \mathcal{X}$, $f(x,u)$ is a convex, continuous function of $u \in \mathcal{U}$ and $\mathcal{U}$ a convex set. The range of such function, for each x, is a convex set.(Prop. 2.32 [16]).

Lipschitz:

The set-valued map is globally Lipschitz at x if there exists a positive constant L such that,

$$\forall x_1, x_2 \in \mathcal{X}, \; F(x_1) \subset F(x_2) + \mathcal{B}_2(0, L||x_1 - x_2||),$$

define, $d(f_1, f_2) := ||f(x_1, u_1) - f(x_2, u_2)||, \forall x_1, x_2 \in \mathcal{X}, \forall u_1, u_2 \in \mathcal{U}$, this implies that[3],

$$\begin{aligned} & d(f_1, f_2) \leq L||x_1 - x_2|| \Rightarrow \\ \Rightarrow\; & d(f_1, f_2)^2 \leq L^2||x_1 - x_2||^2 \\ & \leq L^2(||x_1 - x_2||^2 + ||u_1 - u_2||^2) \Rightarrow \\ & d(f_1, f_2) \leq L||(x_1, u_1) - (x_2, u_2)|| \end{aligned} \tag{10}$$

Since f is C^1, it is a Lipschitz function,

$$\forall x_1, x_2 \in \mathcal{X}, \forall u_1, u_2 \in \mathcal{U},$$

$$||f(x_1, u_1) - f(x_2, u_2)|| \leq M||(x_1, u_1) - (x_2, u_2)||, \tag{11}$$

the constant L, in (10) identifies with M in (11), so there exists a constant such that F is Lipschitz.

Upper Semi-Continuity:

Denote $\mathring{\mathcal{B}}(a, r)$ as the open ball of center a and radius r with some metric. To show USC, assume that $F()$ is not USC everywhere.

Therefore,

$$\begin{aligned} & \exists \bar{x} \in \mathcal{X}, \exists N \supset F(\bar{x}), \forall M(\bar{x}), F(M(\bar{x})) \not\subset N \implies \\ & \implies \forall x \in M(\bar{x}), \forall u \in \mathcal{U}, f(x,u) \notin N \end{aligned} \tag{12}$$

[1] The property of being Lipschitz.

[2] Bolzano-Weierstrass Theorem.

[3] $||(x,y)||^2 = x_1^2 + ... + +x_p^2 + y_1^2 + ... + y_n^2 = ||x||^2 + ||y||^2$

Knowing that x is in a neighborhood of $\bar{x}$,

$$\forall u \in \mathcal{U}, \exists \rho > 0, |(x,u) - (\bar{x},u)| < \rho \implies$$
$$\implies \forall \epsilon > 0, |f(x,u) - f(\bar{x},u)| < \epsilon,$$

that is equivalent to,

$$\forall u \in \mathcal{U}, \forall \epsilon_u > 0, f(x,u) \in \mathring{\mathcal{B}}(f(\bar{x},u), \epsilon_u) \implies$$
$$\implies \forall \epsilon_u > 0, F(x) \subset \bigcup_{u \in \mathcal{U}} \mathring{\mathcal{B}}(f(\bar{x},u), \epsilon_u). \quad (13)$$

To exist a neighborhood N, for $F(\bar{x})$ means that,

$$\exists \epsilon > 0, \forall y \in F(\bar{x}), \mathring{\mathcal{B}}(y, \epsilon) \subset N \Leftrightarrow$$
$$\Leftrightarrow \exists \epsilon > 0, \bigcup_{u \in \mathcal{U}} \mathring{\mathcal{B}}(f(\bar{x},u), \epsilon) \subset N, \quad (14)$$

by the transitive property of inclusions using (13) and (14),

$$\forall x \in M(\bar{x}), F(x) \subset N.$$

This contradicts the proposition in (12), therefore, F must be USC.

Lower Semi-Continuity:

By one hand, admitting that F is not LSC everywhere implies,

$$\exists \bar{x} \in \mathcal{X}, \exists \bar{y} \in F(\bar{x}), \exists N(\bar{y}), \forall M(\bar{x}),$$
$$\exists x \in M(\bar{x}), F(x) \cap N(\bar{y}) = \emptyset \implies$$
$$\forall u \in \mathcal{U}, f(x,u) \notin N(\bar{y}) \implies \nexists \epsilon > 0, |f(x,u) - \bar{y}| < \epsilon. \quad (15)$$

By the other, using the definition of continuity,

$$\forall \bar{x} \in \mathcal{X}, \forall x \in M(\bar{x}), \forall u \in \mathcal{U},$$
$$\exists \rho > 0, |(x,u) - (\bar{x},u)| < \rho \implies$$
$$\implies \forall \epsilon > 0, |f(x,u) - f(\bar{x},u)| < \epsilon,$$

and, as it is valid for any u and, $\bar{y} = f(\bar{x},u)$,

$$\forall \bar{y} \in F(\bar{x}), \exists \epsilon > 0, |f(x,u) - \bar{y}| < \epsilon,$$

this contradicts (15), therefore, F is also lower semi-continuous.
□

Definition 3. (Selection [17]) Consider a set-valued map $F : \mathcal{X} \rightsquigarrow \mathcal{Y}$ with non empty images. A single valued map $f : \mathcal{X} \mapsto \mathcal{Y}$ is called a *selection* of F if for every $x \in \mathcal{X}$, $f(x) \in F(x)$.

The following theorem will be useful ahead in the paper to ensure existence of solutions,

Theorem 2. *([14]) Let F be a LSC set-valued map, from some open set $\Omega \subset \mathbb{R} \times \mathcal{X}$, into some non-empty, closed and convex subset of $\mathcal{Y}$. Let $(0, x_0) \in \Omega$. Then there exists some interval $I = (\omega_-, \omega_+)$, $\omega_- < 0 < \omega_+$ and at least one continuously differentiable function $x : I \mapsto \mathcal{Y}$, where $x(t)$ is a solution to the Cauchy problem[4] for the differential inclusion,*

$$\dot{x}(t) \in F(t, x(t)), \qquad x(0) = x_0.$$

The following set operations will be of great use throughout the paper,

Definition 4. The Minkowski Sum and Pontryagrin Difference between two sets are defined, respectively, as,

$$A \oplus B := \{a + b \mid \in A, b \in B\}$$
$$A \ominus B := \{a \mid a \oplus B \subseteq A\}$$

A ball is a set in some metric space $\mathcal{M}$, with some norm p, defined as,

$$\mathcal{B}_p(x,r) := \{y \in \mathcal{M} \mid ||y - x||_p \leq r\}$$

A. Reachability and Viability

For the proper functioning of th algorithm proposed in this work (Section IV), we have to ensure that the reachable set can be computed and properly define viability kernel. Definitions of both sets are laid out. Afterwards we will explore the conditions of existence of general finite-time reachable sets, independently of the method of discretization or linearization.

Definition 5. The finite time horizon, $T = N_\rho \rho$, reachable set of the continuous non-linear system (4) with piecewise constant input, with initial states in the set $\mathcal{K} \subseteq \mathcal{X}$, is defined as,

$$\text{Reach}_T(\mathcal{K}) := \{x \in \mathcal{X} \mid x = x(T), x_0 \in \mathcal{K} : \exists \{u_0, ..., u_{N_\rho - 1}\} \in \mathcal{U}\} \quad (16)$$

Proposition 1. *Let a function $f : \mathcal{X} \times \mathcal{U} \mapsto \mathcal{Y}$ be defined as in* (5)*, and such that the conditions of Theorem 1 hold. Then, the finite-horizon reachable set,* (16) *can always be computed, for any $T \in \mathbb{R}^+$.*

Proof: From Theorem 1, $F(x)$ is non-empty, with compact, convex values and LSC, therefore we are in the conditions of Theorem 2. This implies the existence of at least one continuously differentiable function, $x(t)$ for some tight interval. Since the properties of $F(x)$ are verified everywhere in its domain, the local existence of solutions becomes global. Therefore, $\forall T \in \mathbb{R}^+$, there exists $x(t)$, $\forall t \in [0, T]$.
□

Proposition 2. *Given a function $f : \mathcal{X} \times \mathcal{U} \mapsto \mathcal{Y}$ defined as in* (5)*. If f is once differentiable in both variables in all its domain, then, for any $x_k \in \mathcal{X}, u_k \in \mathcal{U}, (x^*, u^*) \in \mathcal{X} \times \mathcal{U}$, the step reachable set* (22) *can be computed.*

Proof: Since f is everywhere differentiable, the matrices A and B (defined in (20)) always exist. It is known that the power series defining e^{tA}, converges uniformly on compact time intervals, then A_ρ can be computed. The integral $\int_0^\rho e^{\lambda A} d\lambda$ also converges, so B_ρ and I_ρ exist and can be computed. Since differentiability implies continuity,

[4]Also called initial value problem.

$f(x^*, u^*)$ also exists. Therefore, the step reachable set can be computed always.
□

We will also properly define the main object in the scope of this work,

Definition 6. The finite-horizon sampled-data viability kernel of $\mathcal{K}$, is the set of all the initial states in $\mathcal{K}$ for which there exists at least one trajectory $x_k := x(k\rho)$, with piecewise-constant input, that stays inside the constraint set $\mathcal{K}$ for the time horizon $T = \rho N_\rho$.

$$\text{Viab}_T^{sd}(\mathcal{K}) := \{x_0 \in \mathcal{K} \mid \forall i \in \{0, ..., N_\rho - 1\}, \exists u_i \in \mathcal{U} : x(t) \in \mathcal{K}, \forall t \in [0, T]\}.$$

B. Linearizing Systems

A linear system is described by the following differential equation,

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) + l(t), \quad \forall t > 0 \\ x(t) \in \mathcal{X}, u(t) \in \mathcal{U}, l(t) \in \mathcal{L}, x(0) = x_0. \end{cases} \tag{17}$$

With $\mathcal{X} \subset \mathbb{R}^n$, $\mathcal{U} \subset \mathbb{R}^d$ and $\mathcal{L} \subset \mathbb{R}^n$ compact, convex subsets. In the system above, $l(t)$ is a perturbation of the system, in our case, the linearization error. Define the sampling time as ρ. The time horizon considered is denoted as $\bar{T}$, with the number of time steps being, $N_\rho = \lfloor \bar{T}/\rho \rfloor$. Define the following integral,

$$I_\rho := \int_0^\rho e^{Ar} dr.$$

The sampled-time discrete system becomes,

$$x_{k+1} = A_\rho x_k + B_\rho u_k + I_\rho l_k, \quad \forall k \in \{0, ..., N_\rho - 1\}, \tag{18}$$

with,

$$A_\rho := e^{A\rho}, \; B_\rho := I_\rho B.$$

The state of the system at some time step s, with some initial state x_0, is given by,

$$x_s = A_\rho^s x_0 + \sum_{i=0}^{s-1} A_\rho^i (B_\rho u_{s-i-1} + I_\rho l_{s-i-1}) \tag{19}$$

1) Polytopes: Since the constraint sets are often described as linear inequalities and equalities, a preferred type of representation is the polytopic. There exists two main types of polytopes, the $\mathcal{V}$-polytopes, represented by its vertex and $\mathcal{H}$-polytopes, represented by its facets.

A convex $\mathcal{V}$-polytope, represented by a finite set of points, $\{v_1, v_2, ..., v_k\}$, is the convex hull of such set of points.

$$P = \text{Conv}(\{v_1, v_2, ..., v_k\}).$$

If we stack every linear inequality that defines the polytope we have,

$$P = \{x \in \mathcal{X} \mid Hx \leq b\},$$

where the stacked matrix H and the vector b represent the intersection of all the half spaces (linear inequalities) that define the polytope.

2) Zonotopes: A special class of convex polytopes are the *zonotopes*. They will be the main type of representation for reachable sets in this work. In [18] this representation is fully explained.

Definition 7. (Zonotope)
A zonotope is a set defined in an Euclidean space $\mathbb{R}^n$,

$$Z := \Big\{x \in \mathbb{R}^n \mid x = c + \sum_{i=1}^{p} \alpha_i g_i, \; -1 \leq \alpha_i \leq 1\Big\},$$

where, c is the center of the zonotope, and, $g_1, ..., g_p$ are its generators, all of them are vectors in $\mathbb{R}^n$. Denote the zonotope, Z, as the tuple,

$$Z := (c, < g_1, ..., g_p >).$$

The space of these objects is closed under affine transformations, $\mathcal{A}$, and Minkowski sums.

$$\mathcal{A}Z = (\mathcal{A}c, < \mathcal{A}g_1, ..., \mathcal{A}g_p >,$$

$$Z_1 \oplus Z_2 = (c^{(1)} + c^{(2)}, < g_1^{(1)}, ..., g_p^{(1)}, g_1^{(2)}, ..., g_s^{(2)} >).$$

The fact that the above operations are simple and efficient to implement, they will be our object of election to represent reachable sets.

C. Linearization

The linearization of the system (4) is done using some results in [19], in the following manner,

$$\begin{aligned} \dot{x} \in f(x^*, u^*) &+ \left.\frac{\partial f(x,u)}{\partial x}\right|_{x=x^*, u=u^*} (x - x^*) \\ &+ \left.\frac{\partial f(x,u)}{\partial u}\right|_{x=x^*, u=u^*} (u - u^*) + \mathcal{L} \end{aligned} \tag{20}$$

Matrices $A = \left.\frac{\partial f(x,u)}{\partial x}\right|_{x=x^*, u=u^*}$ and $B = \left.\frac{\partial f(x,u)}{\partial u}\right|_{x=x^*, u=u^*}$ are the Jacobian matrices of the respective linearizations, in terms of state and input variables, around the point (x^*, u^*). The term $\mathcal{L}$ is the Lagrangian Remainder. Let i be the index of the i-th state variable, defining $z = \begin{bmatrix} x \\ u \end{bmatrix}$, the Hessian of the i-th state variable is,

$$J_i(\xi) := \frac{\partial^2 f_i(\xi)}{\partial z^2},$$

the i-th component of the remainder,

$$\mathcal{L}_i = \frac{1}{2}(z - z^*)^T J_i(\xi)(z - z^*).$$

with, $\xi \in \{z^* + \alpha(z - z^*) \mid \alpha \in [0, 1]\}$.

This is the continuous time, linearized system. This system must be converted to a discrete-time version, since that is the representation in the scope of this work. Combining(20) with (18),

$$\begin{cases} \hat{x}_{k+1} = A_\rho^{(k)}\hat{x}_k + B_\rho^{(k)}u_k + \\ \qquad +I_\rho(z^*)(f(z^*) - A^{(k)}x_k^* - B^{(k)}u_k^*) \\ \hat{x}_k \in \mathcal{X}, u_k \in \mathcal{U}, z^* \in \mathcal{X}\times\mathcal{U}, \forall k \in \mathbb{N}^0 \end{cases} \tag{21}$$

The step reachable set of such linearized and discretized system (21), from the set $\mathcal{E}$, around the point z^* is computed as in [20],

$$\text{Reach}_\rho^{lin}(\mathcal{E}) := A_\rho[\mathcal{E}] \oplus B_\rho[\mathcal{U}] + I_\rho(f(z^*) - Ax^* - Bu^*) \tag{22}$$

Usually, it is hard to compute the exact set generated by the Lagrangian remainder. For that matter, an over-approximation is required for it[19].

IV. An Algorithm for approximating Viability Kernels

The main part of the algorithm overlaps with the work done in [21]. However, our algorithm is the result of modifying the feasibility assessment of trajectories, which results in a faster procedure and enabling it to be used for non-linear systems.

A. Basic Algorithm

The original structure of the algorithm, Algorithm 1, is based on sampling a number of random line segments, N_r, starting from a given point v_0 in the viability kernel and ending on the intersection with the boundary of the constraint set $\mathcal{K}$. By means of bisecting that line segment, the algorithm extracts, up to some tolerance, the closest point to the border of the viability kernel, belonging to it. The under approximation is the convex hull, Conv$(\cdot)$, of the points that are guaranteed to be in the viability kernel.

Algorithm 1 [21] Computes a polytopic under-approximation of $\text{Viab}_T^{sd}(\mathcal{K})$

function POLYTOPIC-APPROX($\mathcal{K}, \boldsymbol{v}_0, N_r$)
  $\mathcal{V} \leftarrow \{\boldsymbol{v}_0\}$
  for i from 1 to N_r **do**
    $\boldsymbol{r} \leftarrow$ SampleRay(v_0)
    $\boldsymbol{b} \leftarrow$ IntersectionOnBoundary($\mathcal{K}, \boldsymbol{r}$)
    $\boldsymbol{v}_i \leftarrow$ BisectionFeasibilitySearch($\mathcal{K}, \boldsymbol{b}, \mathcal{K}$)
    $\mathcal{V} \leftarrow \mathcal{V} \cup \{\boldsymbol{v}_i\}$
  end for
  return Conv($\mathcal{V}$)
end function

If $F(x)$ is a non-empty, compact, convex set-valued map and x taking values in a compact convex set $\mathcal{X}$, we assure that the output of Algorithm 1 still works for this system and it is in fact an under-approximation of the viability kernel of the non-linear system.

Also we prove in what conditions does the infinite time viability kernel of a continuous time system (not-sampled), $\dot{x}(t) \in F(x(t))$,

$$\text{Viab}_F(K) := \{x_0 \in K \mid \forall t > 0, \exists u(t) \in \mathcal{U} : x(t) \in \mathcal{K}\},$$

is a convex set. Define the following,

Definition 8. (Kuratowski Set Convergence [22])
Given a sequence of non-empty subsets, $\{A_n\}_{n=1}^\infty$, of a Banach Space X, it converges to A in the sense of *Kuratowski* if,

$$\liminf_{n\to\infty} A_n = \limsup_{n\to\infty} A_n = A,$$

where,

$$\begin{aligned} \limsup_{n\to\infty} A_n = \{x \in X : \exists\{A_{n_k}\}_{k=1}^\infty \text{ of } \{A_n\}_{n=1}^\infty, \\ \exists\{x_k\}_{k=1}^\infty, x_k \in A_{n_k}, \forall k \in \mathbb{N} : \\ \lim_{k\to\infty} x_k = x\}. \end{aligned} \tag{23}$$

And,

$$\begin{aligned} \liminf_{n\to\infty} A_n = \{x \in X : \exists\{x_n\}_{n=1}^\infty, \\ x_n \in A_n, \forall n \in \mathbb{N} : \lim_{n\to\infty} x_n = x\}. \end{aligned} \tag{24}$$

The following lemma will be used in a theorem ahead,

Lemma 1. *Let $\{A_n\}_{n=1}^\infty$ be a sequence of sets, in which $A_{n+1} \subseteq A_n$. Consider the upper limit of the sequence, $\limsup_{n\to\infty} A_n = A$. If A_n is convex, $\forall n \in \mathbb{N}$, then, A is convex.*

Proof: Consider $x, y \in A$ and $\lambda \in [0,1]$. Then, there exist two subsequences $\{A_{n_k}\}_{k=1}^\infty$, $\{A_{\bar{n}_k}\}_{k=1}^\infty$ and two, respective, sequences, $\{x_k\}_{k=1}^\infty$ and $\{y_k\}_{k=1}^\infty$, where, $x_k \in A_{n_k}$, $y_k \in A_{\bar{n}_k}, \forall k \in \mathbb{N}$. With,

$$\lim_{k\to\infty} x_k = x, \quad \lim_{k\to\infty} y_k = y.$$

Since both sequences of elements are convergent, the limit of the convex combination of the sequences is the convex combination of the limits of each sequence,

$$\begin{aligned} \lambda x + (1-\lambda)y &= \lambda \lim_{k\to\infty} x_k + (1-\lambda)\lim_{k\to\infty} y_k \\ &= \lim_{k\to\infty} \lambda x_k + (1-\lambda)y_k. \end{aligned}$$

Choose a new subsequence of sets based on $\hat{n}_k := \min(n_k, \bar{n}_k)$.[5] Since $A_{n+1} \subseteq A_n$, if $\bar{n}_k > n_k$, then, $A_{\bar{n}_k} \subseteq A_{n_k}$, which means that, $y_k \in A_{n_k}$. Remember that by the definition, $x_{n_k} \in A_{n_k}$, therefore, $y_k, x_k \in A_{\hat{n}_k}$. The same applies in the cases where $n_k > \bar{n}_k$. Due to convexity of $A_n, \forall n \in \mathbb{N}$,

$$\lambda x_k + (1-\lambda)y_k \in A_{\hat{n}_k}, \forall k \in \mathbb{N}.$$

[5]This is still a subsequence of $\{A_n\}_{n=1}^\infty$. If $n_k \leq \bar{n}_k$, then, $n_k < \bar{n}_{k+1}$, and reminding that, $n_k < n_{k+1}$, results in, $\hat{n}_k < \hat{n}_{k+1}, \forall k$.

By Definition 8, $\lambda x + (1-\lambda)y \in A$, and since x and y are arbitrary, A is convex.
□

Definition 9. Marchaud Set-Valued Map ([17])

Consider the set valued map, $F : X \rightsquigarrow Y$. Where X, Y are finite dimensional vector spaces. F is Marchaud if,

$$\begin{cases} \text{Dom}(F) \neq \emptyset, \\ F \text{ is USC with convex, compact values,} \\ \forall x \in \text{Dom}(F), ||F(x)|| := \max_{y \in F(x)} ||y|| < c(||x|| + 1), \end{cases}$$

for some $c \in \mathbb{R}$.

Theorem 3. *Let $F : X \rightsquigarrow Y$ be a set-valued map with X, Y finite dimensional vector spaces. If F is a Marchaud, L-Lipschitz set-valued map, satisfying the boundedness condition* (25)*, and $K \subset X$ a compact convex subset, then, the infinite time viability kernel $Viab_F(K)$ is convex.*

$$M := \sup_{x \in K} \sup_{y \in F(x)} ||y|| < \infty \tag{25}$$

Proof: Consider the difference inclusion,

$$x_{n+1} \in \Gamma_\rho(x_n),$$

with, $\Gamma_\rho(x) := x \oplus \rho F(x) \oplus \mathcal{B}(0, \frac{ML}{2}\rho^2)$, $\rho \in \mathbb{R}^+$. The infinite time viability kernel, $\text{Viab}_{\Gamma_\rho}(K)$ can be constructed as [23][24],

$$\begin{cases} K_0 = K, \\ K_{n+1} = K_n \cap \text{Back}_{\Gamma_\rho}(K_n), \end{cases}$$

where, $\text{Back}_{\Gamma_\rho}(K_n)$ is the backward reachable set from K_n - the set of states that end in K_n the next time step. The backward reachable set of Γ_ρ coincides with the reachable set of the system with the inverse dynamics $-F(x)$, $x_{n+1} \in \hat{\Gamma}_\rho(x_n) = x_n \oplus (-\rho F(x)) \oplus \mathcal{B}(0, \frac{ML}{2}\rho^2)$ [11][25]. Since it is a difference inclusion, its reachable set is simply, $\text{Back}_{\Gamma_\rho}(K) = \hat{\Gamma}_\rho(K)$. This is a convex set, since K is convex and F has convex images. In the limit $n \to \infty$,

$$\text{Viab}_{\Gamma_\rho}(K) = K_\infty.$$

By construction, $\text{Viab}_{\Gamma_\rho}(K)$ is a convex set.

Define now a sequence of strictly decreasing values of ρ_n, where, $\lim_{n\to\infty} \rho_n \to 0$. Since $\forall n \in \mathbb{N}, \rho_{n+1} < \rho_n$, $\forall x \in K, \Gamma_{\rho_{n+1}}(x) \subseteq \Gamma_{\rho_n}(x)$, then, $\text{Viab}_{\Gamma_{\rho_{n+1}}}(K) \subseteq \text{Viab}_{\Gamma_{\rho_n}}(K)$.[6] From Theorem 3.2 of [24] and since F is Marchaud, L-Lipschitz bounded by M,

$$\limsup_{\rho \to 0} \text{Viab}_{\Gamma_\rho}(K) = \text{Viab}_F(K).$$

From Lemma 1, $\text{Viab}_F(K)$ is convex.
□

Theorem 4. *Consider the system defined by a differential equation such as* (5)*. Admitting that the state space, $\mathcal{X}$, constraint set, $\mathcal{K}$, and the input set, $\mathcal{U}$, are non-empty, compact convex sets, and $f : \mathcal{X} \times \mathcal{U} \mapsto \mathcal{Y}$ is a C^1 function, convex in the second variable for each fixed x. Given a number of random directions, N_r, and an initial point $\boldsymbol{v}_0 \in Viab_T^{sd}(\mathcal{K})$, and let, $\mathcal{S} = Polytopic\text{-}Approx(\mathcal{K}, \boldsymbol{v}_0, N_r)$, then,*

$$\mathcal{S} \subseteq Viab_T^{sd}(\mathcal{K}).$$

Proof:

Recall that, $\mathcal{S} = \text{Conv}(v_0, v_1, ..., v_{N_r})$. We want to verify that,

$$v_i \in \text{Viab}_T^{sd}(\mathcal{K}), \quad i \in \{0, 1, ..., N_r\} \implies \tag{26}$$

$$\implies \text{Conv}(v_0, v_1, ..., v_{N_r}) \subseteq \text{Viab}_T^{sd}(\mathcal{K}). \tag{27}$$

A sufficient condition for this to hold is the viability kernel to be a convex set. Since we are in the conditions of Theorem 1, we can describe the system by a differential inclusion $\dot{x}(t) \in F(x(t))$, where $F(x)$ has the property (among others) of having non-empty, compact, convex values and being Lipschitz. As in [24] and by the definition of finite time horizon viability kernel of the sampled data system (Def. 6), the finite-time viability kernel for the sampled-data system can be built recursively as,

$$\begin{cases} K_0 = \mathcal{K} \\ K_{n+1} = \{x \in K_n \mid K_n \cap \text{Reach}_\rho(x) \neq \emptyset\}. \end{cases}$$

Where, K_n is the the n step viability kernel. Using a proof from [23],

$$\begin{aligned} x \in K_{n+1} &\Leftrightarrow x \in K_n \wedge (\text{Reach}_\rho(x) \cap K_n \neq \emptyset) \\ &\Leftrightarrow x \in K_n \cap \text{Back}_\rho(K_n) \end{aligned}$$

Notice that the set $\text{Back}_\rho(K_n)$ is the set of initial states in which there are trajectories that end in K_n the next instant, this is the backwards reachability set. This set coincides with the reachable set of the system similar to (4) but with, $\dot{x}(t) \in -F(x(t))$, $x_0 \in K_n$, instead[11][25].

If the step reachable set of the inverse dynamics is convex, then, by construction, $\text{Viab}_T^{sd}(\mathcal{K}) = K_{N_\rho}$ is a convex set.

Definition 10. (Set-Valued Map Composition [26]) Consider two set valued-maps G and H with domains and values in a linear vector space. Define the composition $(G \circ H)(x) := \{z \mid \exists y \in H(x) \text{ with } z \in G(y)$. If a set-valued map $G(x)$ is composed with itself N many times, we denote the resulting set-valued map as $G^N(x)$.

Introducing the following theorem,

Theorem 5. *(Reachable set exponential formula [26])*

Suppose $X \subseteq \mathcal{D} \subseteq \mathbb{R}^n$ is an open set, and $F : \mathcal{D} \rightsquigarrow \mathcal{Y} \subseteq \mathbb{R}^m$ a set valued function. Assume F has non-empty, compact, convex values on X and is locally Lipschitz on X. Then, for all $x \in X$ and $T > 0$, such that all solutions starting at x remain in the set X by time T, we have the reachable set,

$$Reach_T(x) = \lim_{N \to \infty} \left(I + \frac{T}{N} F \right)^N (x),$$

where the limit above defined is in the Kuratowski limit of sequence of sets sense.

[6] Let A, B, S be subsets in the same space, $A \subseteq B \implies A \cap S \subseteq B \cap S$.

Define $G(x) := -F(x)$. Since it was just applied a linear map to the set-valued function, $G(x)$ is still non-empty, compact, convex valued, Lipschitz in $\mathcal{X}$. From the formula of the above Theorem 5, notice that, $\forall x \in \mathcal{K}, \forall N \in \mathbb{N}$, the set $\left(I + \frac{T}{N}G\right)^N(x)$ is convex , because the set valued map $S(x) = I + \frac{T}{N}G(x)$ is convex (linear maps and adding 1 to every element in every dimension of $G(x)$) and the composition of convex imaged set-valued maps on convex-imaged set-valued maps yield convex sets.

From [22], the limit of a sequence of convex sets is convex, therefore, K_n is convex $\forall n \in \mathbb{N}$, and therefore, $\text{Viab}_T^{sd}(\mathcal{K})$ is also convex. That means that $\mathcal{S} \subseteq \text{Viab}_T^{sd}(\mathcal{K})$

□

B. Sampled-Data and error-bounded system

The trajectory of our system, in continuous time, needs to be sampled. That sampling introduces the error of "seeing" the system from time to time. Lemma 2 helps us deal with that so it doesn't affect our computations.

The velocity $\dot{x}(t)$ of the system (4) is bounded for any t, in the subset $\mathcal{X}$. Using the ∞-norm,

$$||\dot{x}(t)||_\infty \leq M, \forall t > 0$$

Lemma 2. *([21]), Consider the system* (4)*, where $\mathcal{X}$ and $\mathcal{U}$ are compact subsets of $\mathbb{R}^n$ and $\mathbb{R}^d$, respectively, M is a ∞-norm bound of $\dot{x}(t)$. The trajectory $x(t)$ is being sampled, with a sampling interval ρ. Define the set*

$$\mathcal{K}^\flat := \mathcal{K} \ominus \mathcal{B}_\infty(0, M\rho). \tag{28}$$

For some initial value x_0, if there exists a sequence of input controls $\{u_0, u_1, ..., u_{N_\rho}\}$ such that, $x(k\rho) \in \mathcal{K}^\flat$, $\forall k \in \{0, ..., N_\rho\}$, then,

$$x_0 \in \text{Viab}_T^{\text{sd}}(\mathcal{K})$$

The proof can be found in [21].

C. Error Bounding

In our case, we would like to take the linearization errors, such as present in (21), into account. Consider the later equation with,

$$e_k = I_\rho(l_k + d_k).$$

The total error of the linearized and discretized system at the instant k, where, l_k is the linearization error and d_k the discretization error. The following Lemma 3 still applies to this.

Lemma 3. *([21]), define $l_k = x_k - \hat{x}_k$ as the linearization error between the the real system state $x_k = x(k\rho)$, and the sampled, linearized model used $\hat{x}_k$. Admit that $||l_k||_\infty \leq \gamma_k$.*

$$\mathcal{K}_k^\flat := \mathcal{K}^\flat \ominus \mathcal{B}_\infty(0, \gamma_k) \neq 0$$

Then,

$$\hat{x}_k \in \mathcal{K}_k^\flat \implies x_k \in \mathcal{K}^\flat.$$

The proof can be found in [21].

If we would like to account for other type of errors, such as truncation and numerical ones, the previous result can be re-used.

D. Modified Algorithm

The procedure adapted for a more general case of non-linear systems is how the feasibility of some point x_0 is determined. The initial condition x_0 of the system is *feasible* in some set if at least one trajectory starting from x_0 stays inside that set. Determining if a point is feasible in a non-linear system framework is done with Algorithm 2.

Algorithm 2 Determines if the evolution of the system with initial set X_0 is feasible at each iteration k in the set $\mathcal{K}_k^\flat$, using reachability.

```
1: function FEASIBLE(N_ρ,ρ,S,X_0,{K_1^♭, ..., K_{N_ρ}^♭})
2:     for k from 1 to N_ρ do
3:         X_k ← Reach_S(ρ[k − 1, k], X_{k−1})
4:         if not Overlaps(X_k, K_k^♭) then
5:             return False
6:         end if
7:     end for
8:     return True
9: end function
```

Proposition 3. *Consider a point $x_0 \in \mathcal{X}$, if* FEASIBLE$(N_\rho, \rho, \mathcal{S}, x_0, \{\mathcal{K}_1^\flat, ..., \mathcal{K}_{N_\rho}^\flat\})$ *returns* True*, then $x_0 \in \text{Viab}_T^{\text{sd}}(\mathcal{K})$, with $T = \rho N_\rho$.*

Proof: From the algorithm returning True we have,

$$\forall k \in \{1, ..., N_\rho\}, X_k \cap \mathcal{K}_k^\flat \neq \emptyset \Rightarrow$$
$$\forall s \in \{0, ..., N_\rho - 1\}, \exists u_s \in \mathcal{U} : \hat{x}_{s+1} \in \mathcal{K}_{s+1}^\flat. \tag{29}$$

Using Lemma 3, and rearranging the quantifiers, (29) implies that,

$$\forall s \in \{1, ..., N_\rho\}, \exists \{u_0, ..., u_{N_\rho - 1}\} : x_s \in \mathcal{K}^\flat.$$

Which, by Lemma 2,

$$x_0 \in \text{Viab}_T^{sd}(\mathcal{K}).$$

□

Under-approximations of the reachable set can also be used, as shown in the following proposition,

Proposition 4. *Consider Algorithm 2, with line 3 replaced by the under-approximation of the reachable set,*

$$\overline{X}_k \leftarrow \mathcal{P}_k \subseteq \text{Reach}_\mathcal{S}(\rho[k-1, k], X_{k-1}), \quad \forall k = \{0, 1, ..., N_\rho\}$$

If $\overline{X}_k \cap \mathcal{K}_k^\flat \neq \emptyset$, then, we are able to construct an under-approximation of the originally intended viability kernel,

$$x_0 \in \text{Viab}_T^{sd,\flat}(\mathcal{K}) \subseteq \text{Viab}_T^{sd}(\mathcal{K}).$$

Proof: By proposition 3,

$$\forall k \in \{1, ..., N_\rho\}, \overline{X}_k \cap \mathcal{K}_k^\flat \neq \emptyset \Rightarrow x_0 \in \text{Viab}_T^{sd,\flat}(\mathcal{K}). \tag{30}$$

Here $\text{Viab}_T^{sd,\flat}(\mathcal{K})$ represents the viability kernel of the system whose step reachability is $\overline{X}_k$. Also, from the original reachability, $X_k \leftarrow \text{Reach}_{\mathcal{S}}(\rho[k-1,k], X_{k-1})$, with $\overline{X}_k \subseteq X_k$, (30) implies that,

$$x_0 \in \text{Viab}_T^{sd}(\mathcal{K}).$$

Consider now the case where,

$$X_k \cap \mathcal{K}_k^{\flat} \neq \emptyset,\ \overline{X}_k \cap \mathcal{K}_k^{\flat} = \emptyset.$$

By prop. 3, $x_0 \in \text{Viab}_T^{sd}(\mathcal{K})$, but, $x_0 \notin \text{Viab}_T^{sd,\flat}(\mathcal{K})$

By the definition of set inclusions,

$$\text{Viab}_T^{sd,\flat}(\mathcal{K}) \subseteq \text{Viab}_T^{sd}(\mathcal{K}).$$

□

E. Computing reachable sets of non-linear systems

Propositions 1 and 2, establish the condition in which it is possible to compute the reachable set. The assurance of these conditions is very useful if one wants to guarantee that the algorithm terminates with a definite answer and do not end with an ill-posed problem.

The reachable set computation using linearizations (Algorithm 3) receives as inputs the discretization step ρ, the system's non-linear model, $\mathcal{S}$, the input space, $\mathcal{U}$, the initial condition set of the system, X_0, and the admissible Lagrangian remainder $\bar{\mathcal{L}}$.

Algorithm 3 Computes the step reachability set of non-linear systems [19]

function REACH(ρ,$\mathcal{S}$,$\mathcal{U}$,X_0,$\overline{\mathcal{L}}$)
  $(A, B, \mathcal{L}) \leftarrow$ Linearize($\mathcal{S}, X_0$)
  if $\mathcal{L} \not\subseteq \overline{\mathcal{L}}$ **then**
    $(X_0^{(1)}, X_0^{(2)}) \leftarrow$ Split(X_0)
    $R^{(1)} \leftarrow$ Reach($\rho, \mathcal{S}, \mathcal{U}, X_0^{(1)}, \overline{\mathcal{L}}$)
    $R^{(2)} \leftarrow$ Reach($\rho, \mathcal{S}, \mathcal{U}, X_0^{(2)}, \overline{\mathcal{L}}$)
    return $\{R^{(1)}, R^{(2)}\}$
  else
    $R \leftarrow AX_0 \oplus B\mathcal{U}$
    return $\{R\}$
  end if
end function

As previously stated, the zonotopes are the preferd type of set representation, since linear maps and Minkowski Sums are efficiently computed. Since the union of zonotopes is not necessarily a zonotope, and it is not necessary the explicit computation (although it might be rendered useful for tight admissible linearization errors) one avoids doing the explicit computation, and simply add them in a list.

Before proceeding to the computation of the viability kernel, the user must provide the limit for the admissible Lagrangian remainder, $\bar{\mathcal{L}}$, before the splitting takes place. The admissible bound for the remainder is used in eroding the constraint set as in Lemma 3.

The splitting of zonotopes is done in the following way,

Proposition 5. *([19]) (Splitting of Zonotopes) A zonotope, $Z = (c, < g_1, ..., g_p >)$, can be split into two zonotopes Z_1 and Z_2 such that $Z_1 \cup Z_2 = Z$, where,*

$$Z_1 = (c - \frac{1}{2}g_j, < g_1, ..., g_{j-1}, \frac{1}{2}g_j, g_{j+1}, ..., g_p >)$$
$$Z_2 = (c + \frac{1}{2}g_j, < g_1, ..., g_{j-1}, \frac{1}{2}g_j, g_{j+1}, ..., g_p >)$$

F. Overlap between a zonotope and a $\mathcal{H}$-polytope

In order to determine if there are trajectories that keep the system inside the constraint set, the test for overlap between the reachable set and the constraint set must be performed. Using the aforementioned definition of zonotope, one will pose this problem as a simple feasibility program. The program will consist in determining the existence of the coefficients α_i such that the zonotope generated respects the inequality constraints derived from the facet representation of the constraint set. The inequalities from the constraint set are in the form of,

$$H\boldsymbol{x} \leq b. \tag{31}$$

Defining the matrix where the generators of the zonotope defining the reachable set are its columns and the vector of the corresponding coefficients,

$$G = \begin{bmatrix} \boldsymbol{g_1} & \cdots & \boldsymbol{g_p} \end{bmatrix}, \qquad \boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 & \cdots & \alpha_p \end{bmatrix}^T$$

Substituting x in (31) by the definition of zonotope, with column-vectors, yields,

$$H(\boldsymbol{c} + G\boldsymbol{\alpha}) \leq b$$

This is equivalent to the standard notation of inequalities for optimization with $\boldsymbol{\alpha}$ as the variable to optimize,

$$HG\boldsymbol{\alpha} \leq b - H\boldsymbol{c} \Leftrightarrow H^*\boldsymbol{\alpha} \leq b^*.$$

Other restrictions have to be added since each α_i only assumes values in $[-1, 1]$, so, the optimization problem becomes,

$$\begin{array}{rrl} \min\limits_{\boldsymbol{\alpha}} & 0 & \\ \text{s.t.} & H^*\boldsymbol{\alpha} & \leq b^* \\ & I\boldsymbol{\alpha} & \leq 1 \\ & -I\boldsymbol{\alpha} & \leq 1 \end{array} \tag{32}$$

The Overlaps($\cdot$) function, returning whether a zonotope Z and a polytope P intersect each other or not, is then defined as,

$$\text{Overlaps}(Z, P) = \begin{cases} \text{True}, & \exists \boldsymbol{\alpha} \text{ in (32)} \\ \text{False}, & \text{o.w.} \end{cases}$$

For each feasible point computation, it is solved *at most* a $N_\rho p$ dimensional problem with $(h+2)N_\rho p$ constraints, where h are the number of facets of the polytope representing the constraint set and p, the number of generators of the zonotope. The number of computations, for a point x_0 that is not feasible, is largely reduced, for instance, if the first time steps computing each of the reachability set do not overlap with the constraint set, the algorithm terminates.

V. RESULTS

This section presents simulation results that point to the correctness of the algorithm. For the sake of simplicity, we will restrict our experiments to two-dimensional systems. The system being tested is a simple pendulum on a cart, with the adequate dimensions, where x_1 is the pendulum angle from the upright position and x_2 its angular velocity. The system is described by the following non-linear differential equation.

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \sin x_1 - u \cos x_1 \end{cases} \tag{33}$$

Here $x = (x_1, x_2) \in \mathcal{X} \subset \mathbb{R}^2$ and $u \in \mathcal{U} = [-1, 1]$. The goal here is to determine the set of admissible pairs of angular position and angular velocity for the pendulum be kept inside the constraint set, $\mathcal{K} = [-\pi/6, \pi/6] \times [-\pi/6, \pi/6]$, using the adapted algorithm for non-linear systems. It was set $\rho = 0.05$, the number of time steps, $N_d = 10$, the number of sampling rays was set to $N_r = 100$, the bisection accuracy $\epsilon = 0.03$ and the linearization tolerance, $\max_{x \in \bar{\mathcal{L}}} |x| = 0.03$.

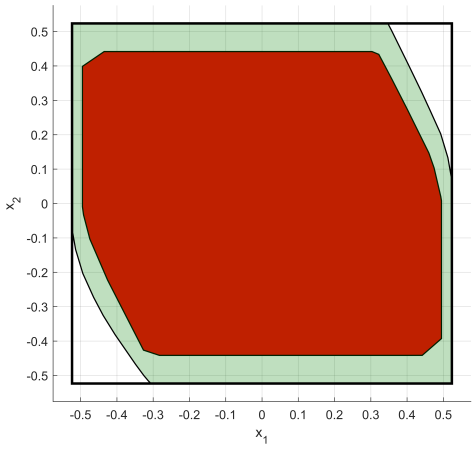

Fig. 4. The outer line is the limit of the constraint set, the light colored (green) set is the finite horizon viability kernel with Saint-Pierre algorithm and the dark color (red) is the kernel computed with our adaptation.

Figure 4 shows two viability kernels computed using two different methods. The one in lighter color is computed using Saint-Pierre's algorithm. As in [24], this algorithm can also return the finite-time viability kernel of the system. For that algorithm, $\rho = 0.05$, the space discretization step $h = 0.0033$, and the Lipschitz constant was set to $L = 2$, and it stopped at the $10th$ iteration so as to compute the 10-step viability kernel. The viable states computed by our algorithm are strictly contained in the one computed with the griding method. Since the accuracy of our method is user defined by the bisection accuracy, ϵ, and the acceptable Lagrangian remainder, $\overline{\mathcal{L}}$, our solution can approximate the true $\text{Viab}^{sd}(\mathcal{K})$ as close as one would want, at the expense of more computations. Since the outer one is not constrained by having piecewise constant inputs, it always includes the one that is constrained.

A. Comments

This section of the present thesis described an algorithm that adapts Gillula's *et al.* work on computing viability kernels for non-linear, sampled systems by linearizing and discretizing. We also prove the algorithm's correctness. The approach presented here reduces the finite horizon viability kernel computation problem to an iterative reachability one. With the appropriate choice of representation for sets, such as zonotopes and polytopes, we used the same algorithms from linear systems to non-linear ones.

As noted in section V, our algorithm under-approximates the true finite horizon viability kernel. In Figure 4 we can compare the infinite horizon viability kernel produced by Saint-Pierre's algorithm with ours. As it is known, for every time step, ρ, the viability kernel computed with Saint-Pierre algorithm is an under-approximation of the true viability kernel for the continuous time system. A system with piecewise constant input will have less viable trajectories than one that can change inputs between sampling intervals (since it is a subset), so, $\text{Viab}_T^{sd}(K) \subseteq \text{Viab}_T(K)$, where the latter set is the finite-time viability kernel for the continuous time system (not sampled). The results confirm this hypothesis, allowing us to confirm the correctness of the algorithm by simulation.

For a sufficiently large linearization error tolerance - does not require many set splitings - the proposed algorithm has a lower asymptotic complexity than that of strategies using state space griding, as in Saint-Pierre's type of methods, which is $\mathcal{O}(a^n)$, for some constant a. In these cases our algorithm has a complexity similar to the one proposed in [21], greater than $\mathcal{O}(n^2)$ but smaller than $\mathcal{O}(n^3)$, since the most complex operations involve running feasibility programs (the overlap test between sets).

There are essentially two points to be careful when employing our algorithm. One is that methods similar to ours only yield finite horizon viability kernels. It is not possible to guarantee viable trajectories in infinite time, since for that the algorithm had to run forever. This is not a limitation from an application perspective. The other one is on the linearization errors. If one is largely tolerant on the linearization error, our under-approximation might be too conservative or even result in the empty-set due to the erosion process. If the tolerance is excessively strict, this may result in an enormous amount of splittings of sets, making the algorithm much slower.

Further improvements can be expected if better representations are found or better ways to compute the overlap between sets were used. Work on how to compute tight bounds for the Lagrangian Remainder is of great importance as well. The performance of the algorithm would be greatly improved if clever ways of splitting and storing the splitted sets were introduced.

Since the scope of this work is in the field of autonomous vehicles, this algorithm can be directly applied for sufficiently complex models. The computation of the viability kernels will be the basis in the synthesis of viable controls that keep the vehicle in safe states.

VI. Conclusions

In this thesis we've presented an architecture of control for an autonomous vehicle, which can be used with any autonomous mobile robot. This architecture was inspired in how humans drive and introduced the sense of feedback on the uncertainty of the task. This type of feedback, adjusting the planning based on viability theory represents a novelty in this field. This controller makes the whole system hybrid, since that at each time step, the dynamics of the vehicle change.

The main focus of this work was in the development of the local planning module, which is based in computing the viability kernel. We've adapted an existing algorithm for the linear system, recurring to reachability, represented with zonotopes. Our algorithm was faster, for a large range of variables, than the original one. Although we didn't prove it was asymptotically faster, it was not disproven either - based on fitting the computational time as a function of the system dimension $n = \{5, 6, ..., 100\}$ with a confidence level of 95%. We've extended the algorithm to be used in some non-linear systems (the majority of the ones with practical relevance), proved its correctness and provided formulas to over-approximate the linearization errors. We claim that in some conditions, sufficiently large admissible linearization errors, our algorithm outperforms most methods based on griding the state-space. The case of the convexity of the viability kernel for continuous time and sampled time systems was also addressed with the adequate proofs.

VII. Future Work

Along this project, there were many uncharted paths one could pursue with the many ideas that occurred. Here are a few of them with a brief explanation,

A. Develop the remaining Architecture

The most obvious next step is to develop the conceptual blocks of the architecture here proposed.

- *Sensor and Perception* block would greatly benefit from an intense study of the minimal optimal amount of sensors and what type they need to be, while aiming also at the lowest cost possible. One interesting idea to implement in the perception layer is to detect the pedestrian awareness of the ego-vehicle, see a related case [27]. For example, the ego-vehicle is near a crosswalk and a pedestrian is not facing the vehicle, he/she might go straight to the crosswalk without even looking for cars. If trajectories are chosen to take into account this occurrences, a few more accidents could be avoided.
- As told in the respective chapter, methods to find optimal paths in graphs are exhaustively present in the literature, picking one to implement *Map-Based Planning* block should not be troublesome.
- Properly define the stage cost functions and the respective minimization algorithms for the controller present in the *Reflexive Planning*.
- Contingency plans in case of failures in each block. What measures should be employed in case the viable set of inputs is the empty-set. What to do in case an unexpected obstacle appears.

B. Simulation the full model and implementation in a real system

After developing the remaining blocks of the architecture it is necessary to simulate the system, for the sake of having an idea of the performance of the overall architecture. Simulation will allow to fine tune some parameters and hierarchical relations among the several modules of the system.

The final step is to implement in a real system. This is almost always the most cumbersome step when developing a theoretical algorithm. Sampling times, parameters of the actual controllers of the actuators are some examples of important variables to take into account in the real system.

C. Non-Convex Constraint Sets

One interesting problem is in the case of non-convex constraint sets. The obvious next step is to apply convex decomposition algorithms to that set, and convert them to a list of convex polyhedron, for example. However, there is the subtlety on how to deal with the boundaries between divisions of the set. We propose the idea of hard and soft boundaries when applying the algorithms presented here. Hard boundary is the one that separates the constraint set from the rest of the space. The soft ones separate the convex divisions of the original constraint set. This adds complexity on is currently done, since the algorithm has to make the overlap test with all the remaining sub-parts of the constraint set.

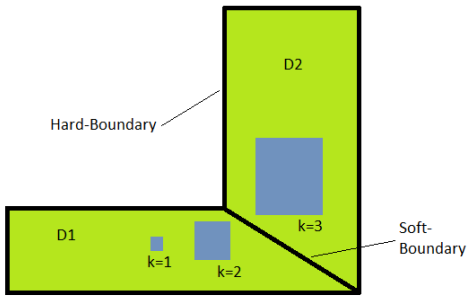

Fig. 5. The reachable sets in dark (blue) color, the non-convex constraint set in light (green) color and its divisions

D. Dynamic Viability Kernels

What if the constraint sets have their own dynamics, despite of the system behavior, how would that influence the viability kernel. It would be reasonable to assume that if one knows how the constraint set changes in time, we would be able to modify the viability kernel accordingly, without having to compute it all over again. This particular question could introduce a real novelty since, to the best of our knowledge, there are no theoretical results to this problem.

REFERENCES

[1] A. Kurkin, D. Tyugin, V. Kuzin, A. Chernov, V. Makarov, P. Beresnev, V. Filatov, and D. Zeziulin, "Autonomous mobile robotic system for environment monitoring in a coastal zone," *Procedia Computer Science*, vol. 103, pp. 459 – 465, 2017, xII International Symposium Intelligent Systems 2016, INTELS 2016, 5-7 October 2016, Moscow, Russia. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050917300236

[2] P. R. Wurman, R. D'Andrea, and M. Mountz, "Coordinating hundreds of cooperative, autonomous vehicles in warehouses," *AI magazine*, vol. 29, no. 1, p. 9, 2008.

[3] J. V. Brummelen, M. OBrien, D. Gruyer, and H. Najjaran, "Autonomous vehicle perception: The technology of today and tomorrow," *Transportation Research Part C: Emerging Technologies*, vol. 89, pp. 384 – 406, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X18302134

[4] A. Naweed and J. Rose, "Assessing technology acceptance for skills development and real-world decision-making in the context of train driving," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 52, pp. 86 – 100, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1369847817306769

[5] K. Colonna, "Autonomous cars and tort liability," *Journal of Law, Technology & the Internet*, 2013.

[6] D. J. Fagnant and K. Kockelman, "Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations," *Transportation Research Part A: Policy and Practice*, vol. 77, pp. 167 – 181, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0965856415000804

[7] C. Urmson and W. . Whittaker, "Self-driving cars and the urban challenge," *IEEE Intelligent Systems*, vol. 23, no. 2, pp. 66–68, March 2008.

[8] D. Howard and D. Dai, "Public perceptions of self-driving cars: The case of berkeley, california," in *Transportation Research Board 93rd Annual Meeting*, vol. 14, no. 4502, 2014.

[9] M. A. Nees, "Acceptance of self-driving cars: an examination of idealized versus realistic portrayals with a self-driving car acceptance scale," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 60, no. 1. SAGE Publications Sage CA: Los Angeles, CA, 2016, pp. 1449–1453.

[10] G. J. S. Wilde, "The theory of risk homeostasis: Implications for safety and health," *Risk Analysis*, vol. 2, no. 4, pp. 209–225, 1982. [Online]. Available: http://dx.doi.org/10.1111/j.1539-6924.1982.tb01384.x

[11] J. Aubin, A. Bayen, and P. Saint-Pierre, *Viability Theory: New Directions*, ser. Modern Birkhèauser classics. Springer Berlin Heidelberg, 2011. [Online]. Available: https://books.google.pt/books?id=0YpZNVBXNK8C

[12] A. Dontchev and F. Lempio, "Difference methods for differential inclusions: A survey," *SIAM Review*, vol. 34, no. 2, pp. 263–294, 1992. [Online]. Available: https://doi.org/10.1137/1034050

[13] G. V. Smirnov, *Introduction to the theory of differential inclusions*. American Mathematical Society, 2002.

[14] J. P. Aubin and A. Cellina, *Differential inclusions: set-valued maps and viability theory*. Springer-Verlag Berlin Heidelberg, 1984.

[15] K. A. Ross, *Elementary Analysis: The Theory of Calculus*, 2013.

[16] R. T. Rockafellar and R. J.-B. Wets, *Variational analysis*. Springer, 1998.

[17] J. Aubin and H. Frankowska, *Set-Valued Analysis*. Birkhuser Basel, 1990.

[18] A. Girard, "Reachability of uncertain linear systems using zonotopes," *Hybrid Systems: Computation and Control Lecture Notes in Computer Science*, p. 291305, 2005.

[19] M. Althoff, "Reachability analysis and its application to the safety assessment of autonomous cars (phd thesis)," Ph.D. dissertation, Technische Universitt Mnchen, 07 2010.

[20] A. Girard, C. Le Guernic, and O. Maler, "Efficient computation of reachable sets of linear time-invariant systems with inputs," in *Hybrid Systems: Computation and Control*, J. P. Hespanha and A. Tiwari, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 257–271.

[21] J. H. Gillula, S. Kaynama, and C. J. Tomlin, "Sampling-based approximation of the viability kernel for high-dimensional linear sampled-data systems," *Proceedings of the 17th international conference on Hybrid systems: computation and control - HSCC 14*, 2014.

[22] S. W. P. L. Papini, "Nested sequences of sets, balls, hausdor convergence," *Note di Matematica Volume 35, Issue 2*, 2015.

[23] J. N. Maidens, S. Kaynama, I. M. Mitchell, M. M. Oishi, and G. A. Dumont, "Lagrangian methods for approximating the viability kernel in high-dimensional systems," *Automatica*, vol. 49, no. 7, pp. 2017 – 2029, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109813001891

[24] P. Saint-Pierre, "Approximation of the viability kernel," *Applied Mathematics & Optimization*, vol. 29, no. 2, p. 187209, 1994.

[25] S. Raczynski, "Uncertainty, dualism and inverse reachable sets," *International Journal of Simulation Modelling*, vol. 10, no. 1, pp. 38–45, 2011.

[26] P. R. Wolenski, "The exponential formula for the reachable set of a lipschitz differential inclusion," *SIAM Journal on Control and Optimization*, vol. 28, no. 5, pp. 1148–1161, 1990. [Online]. Available: https://doi.org/10.1137/0328062

[27] E. A. Sisbot, L. F. Marin-Urias, R. Alami, and T. Simeon, "A human aware mobile robot motion planner," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 874–883, Oct 2007.