Unifying Control and Verification of Cyber-Physical Systems (UnCoVerCPS)

WP2 Online Decision Making and Control (Task 2.1)

D2.1 – Report on Behaviour Prediction for Cyber-Physical Systems

WP2	**D2.1 – Report on Behaviour Prediction for Cyber-Physical Systems**
Authors	Matthias Althoff - Technische Universität München Aaron Pereira - Technische Universität München Silvia Magdici - Technische Universität München Bastian Schürmann - Technische Universität München Goran Frehse - Université Joseph Fourier Grenoble 1 Geoff Pegman - R.U.Robots Limited
Short Description	We present novel techniques for the set-based prediction of surrounding entities in automated driving and human-robot co-working. In automated driving, surrounding entities are traffic participants, while in human-robot co-working, surrounding entities are humans. Reachability analysis is used to predict the set of all possible future behaviors.
Keywords	Behavior prediction, abstract models, reachability analysis, automated driving, human-robot coexistence and collaboration.
Deliverable Type	Report
Dissemination level	Public
Delivery Date	31 Dec 2015
Contributions by	—
Internal review by	Jens Oehlerking - Robert Bosch GmbH Javier Sanchez - Tecnalia
External review by	—
Internally accepted by	Matthias Althoff
Date of acceptance	18 Dec 2015

Document history:

Version	**Date**	**Author/Reviewer**	**Description**
1.0	01 Oct 2015	Althoff et al.	Internal review version
1.1	18 Dec 2015	Althoff et al.	Final version

Contents

1 Introduction

UnCoVerCPS develops new methods and tools for cyber-physical systems with (fully or partially) autonomous behavior in partially known and time-varying environments. Since the environment is dynamic and only partially known, constraints for the controller also become uncertain and time-varying, which makes them only fulfillable during the operation of the system. Therefore, we cannot design a controller a-priori. Instead, we rely on automatically generated constraints during runtime. Especially in our application areas automated driving and human-robot cooperation, constraints for controller design are hard to obtain since the respective environments are populated with intelligent agents whose intentions are unknown. In automated driving, it is not clear what the future actions of surrounding traffic participants are and in human-robot cooperation, it is not obvious what the future actions of surrounding humans are.

This deliverable presents new techniques that compute all possible future behaviors of surrounding intelligent agents, especially those of traffic participants in road traffic and human workers in a factory, based on a dynamical model. The set of all possible behaviors makes it possible to obtain the future occupancy of those agents over time in space. Those occupancy regions, which grow over time due to the uncertainty in the future behavior, are used as constraints for the online controller synthesis. When the controller design can ensure that it never enters any occupancy of surrounding agents, it can be proven that the overall system behavior is free of collisions. In order to prove this property, we compute future possible occupancies in an over-approximative way, i.e. the true occupancy is guaranteed to be found in the computed over-approximation.

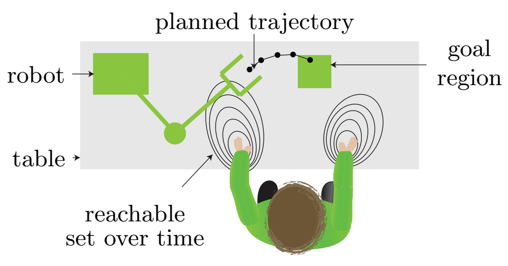


Figure 1: Human worker prediction.

The process of predicting future behavior is constantly repeated using new measurements of the environment. The set of future behaviors is obtained from reachability analysis techniques, which are developed in UnCoVerCPS. We also consider uncertain measurements by enlarging measured states by the set of possible measurement errors. In Fig. 1, the behavior prediction via reachability analysis is sketched for our application scenario *human-machine collaborative manufacturing*. The predicted occupancy of the arms of a human worker over time is used as a forbidden region for the control design, providing the constraints of the controller.

2 Over-Approximative Computations

We first introduce reachability analysis to obtain the set of possible behaviors. We denote the bounded set of uncertain initial states as $\mathcal{R}(0)$. Uncertain initial states are important since it is difficult to exactly measure the state of surrounding entities. If not all states can be directly measured, one requires state observers. Those are not exact as well and thus, assuming a set of possible initial states makes it possible to consider all eventualities. Further, we introduce the bounded set $\mathcal{U}$ of possible inputs to the model of the agents, modeling possible actions of the agent. The solution to its dynamical model $\dot{x} = f(x, u)$ for $x(0) = x_0$, $t \in [0, t_f]$, and input trajectory $u(\cdot)$ is denoted by $\chi(t, x_0, u(\cdot))$. Note that $u(\cdot)$ refers to a trajectory, whereas $u(t)$ refers to the value of the trajectory at time t. The exact reachable set for an uncertain initial set $\mathcal{R}(0)$ and a set of possible inputs $\mathcal{U}$ is

$$\mathcal{R}^e([0, t_f]) = \Big\{\chi(t, x_0, u(\cdot))\Big| t \in [0, t_f],\, x_0 \in \mathcal{R}(0),\, \forall t:\, u(t) \in \mathcal{U}\Big\}. \tag{1}$$

In general, the set of reachable states cannot be computed exactly [1], so that one has to compute over-approximations $\mathcal{R}([0, t_f]) \supseteq \mathcal{R}^e([0, t_f])$.

However, since the reachable set has to be obtained in a very short amount of time due to the online computation of constraints, a direct application of reachability analysis techniques is often not feasible. Instead, we compute abstractions of the dynamics of other agents and intersect the resulting reachable sets. This procedure is over-approximative as shown in Proposition 2.1. The abstractions additionally make it possible to exploit special properties such as monotonicity as later discussed in detail for the applications *automated driving* and *human-robot interaction*. For combining the results of the abstractions, we introduce an operator reach() returning the reachable set of a model M_i for an infinite time horizon, and the projection operator proj(), which projects a set onto dimensions relevant for occupancy checks (typically position and orientation).

Proposition 2.1 (Overapproximative Occupancy) *Given are models M_i, $i = 1 \dots m$ which are abstractions of model M_0, i.e.,* $\mathrm{reach}(M_0) \subseteq \mathrm{reach}(M_i)$. *The occupancy of the model M_0 can be overapproximated by*

$$\mathrm{proj}\big(\mathrm{reach}(M_0)\big) \subseteq \bigcap_{i=1}^{m} \mathrm{proj}\big(\mathrm{reach}(M_i)\big). \qquad \square$$

Proof 1 *Since* $\mathrm{reach}(M_0) \subseteq \mathrm{reach}(M_i)$*, we have that*

$$\begin{aligned} \mathrm{reach}(M_0) &\subseteq \bigcap_{i=1}^{m} \mathrm{reach}(M_i) \\ \rightarrow \mathrm{proj}\big(\mathrm{reach}(M_0)\big) &\subseteq \mathrm{proj}\Big(\bigcap_{i=1}^{m} \mathrm{reach}(M_i)\Big) \end{aligned}$$

Further, it is shown in [2, Prop. 1] that

$$\mathrm{proj}\Big(\bigcap_{i=1}^{m} \mathrm{reach}(M_i)\Big) \subseteq \bigcap_{i=1}^{m} \mathrm{proj}\big(\mathrm{reach}(M_i)\big).$$

The proposed divide and conquer approach is much faster than directly applying reachability analysis since the computational complexity of reachability analysis is superlinear, thus the added computational time of partial reachability problems is typically much less than solving the problem at once.

3 Set-Based Prediction of Road Vehicles for Automated Driving

3.1 Previous Work

Automated cars are expected to provide a broad range of benefits compared to human-driven vehicles. Among them are an increased road safety, better traffic throughput, and increased mobility for people who cannot drive. Today, 90% of car crashes are caused by human error [3], which could be prevented by fully or partially automated vehicles.

Collision avoidance systems are an important component in the architecture of vehicles for making driving safer [4]. Collision avoidance systems could either override decisions of humans or software components of automated vehicles. We envision collision avoidance systems that are provably correct and certifiable. Due to formal correctness, they should have the approval to override decisions, even those of other decision making components in automated cars, which are typically not formally correct. In order to guarantee safe emergency maneuvers, the occupancy for other traffic participants has to be predicted as shown in Fig. 2.

We group previous work on occupancy prediction into four main categories: approaches computing a single future behavior, a countable set of possible behaviors, probability distributions of future behaviors, and uncountable sets of possible behaviors.

Previous work that predicts only a single behavior of other traffic participants can be found in [5], [6], [7], and [8]. In [9] the prediction of the behavior of surrounding vehicles is

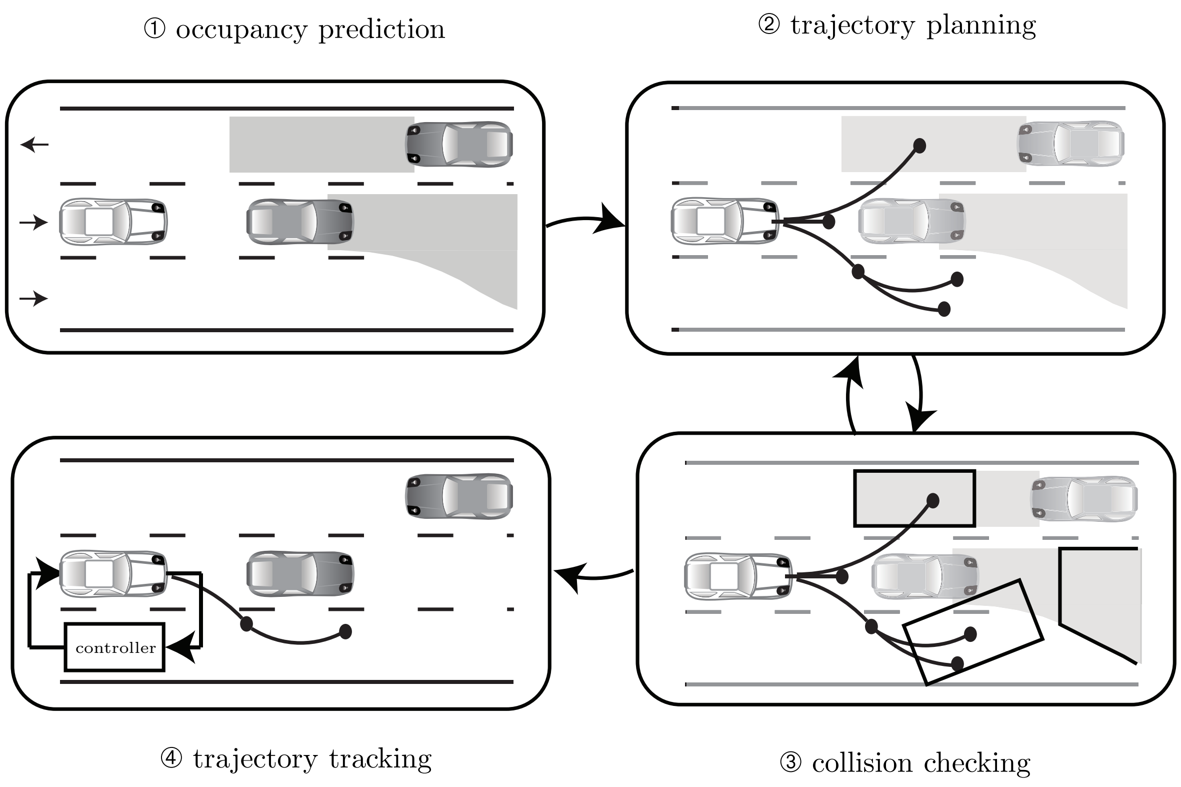


Figure 2: Collision avoidance concept.

done under the assumption that they tend to keep their current velocity and distance to their leading vehicle constant, which is not necessarily correct. Also the focus of their method is in-lane driving, without considering complex situations. Thus, this approach might miss critical scenarios due to the fact that many more future behaviors could result in an accident.

The next class of methods considers a countable set of possible trajectories of other traffic participants. Simulation approaches are widely used for collision assessment. In [10] multiple simulations have been used to generate different physically possible trajectories. Another simulation-based method uses Monte Carlo sampling, where the trajectories are randomly generated, as in [11] and [12]. However, since the number of traffic scenarios are infinitely many, simulation techniques cannot guarantee safety.

Since many possible future behaviors exist, a group of works predicts the probability distribution of future behavior. In [13] the behavior of other traffic participants is predicted using Dynamic Bayesian Networks (DBN). The models that describe the traffic prediction as a DBN in [13] are learned online from unlabeled traffic observations. A map-based prediction is proposed in [14], based on a stochastic filter able to determine the set of reasonable future trajectories. The authors of [15] propose a probabilistic prediction algorithm of a generic driver intent, capable of separating the straight driving from a right-turn maneuver. However, this approach relies on accurate measurement of the driver's gaze direction, which is often difficult to obtain. Another probabilistic approach has been suggested in [16] where the probability of a crash is assessed. The probability distribution of a collision is modeled using a product of Gaussians. Due to the probabilistic nature of the aforementioned approaches, they cannot guarantee safety for bounded uncertainties unless probability distributions can be exactly computed for very simple scenarios. In [17] and [18] the dynamics of traffic participants is abstracted to Markov Chains in order to perform safety assessment.

To avoid any collision, every planned maneuver needs to be checked against the occupancy prediction of other traffic participants. While some maneuvers can be precomputed [19], e.g. turn left, turn right etc, the collision check must be performed constantly, since every traffic situation is unique and a pre-computation of discretized situations is prohibitively large. As computation time constraints must be fulfilled, most of the previous work in mobile robotics considers a very simple model of surrounding dynamic obstacles. One of the simplest models used for behavior prediction considers a point mass with limited velocity or acceleration [20], [21]. Other non-holonomic models can also be used, like Dubin's car [22], or a tricycle model [23]. However, for complicated models, which are taking into account the road boundaries or possible lane changes, a fast method is not known.

In order to guarantee safety, a Model Predictive Control framework can be used to incorporate the constraints based on the state of other vehicles in the proximity of the ego car [24], [25]. Although the proposed theoretical approach from [24] is provably safe, the authors assume that the communication between the vehicles is without error and information such as lane change intention is perfectly known. In reality, these assumptions are very hard to achieve. Recently, the concept of potential fields has also been applied for driving safety assessment [26]. However, this approach considers full information access to model parameters of other traffic participants.

In the architecture proposed in [27], the importance of trajectory prediction of other vehicles is highlighted, but no algorithm for formally computing the occupancy prediction is suggested. In this deliverable, a framework for robust and fast occupancy prediction of automated vehicles is proposed. The approach makes use of reachability analysis, hence the predicted occupancy sets are guaranteed safe, taking into consideration all possible trajectories.

3.2 Overview and Assumptions

Our approach is based on a module that recurrently checks whether collision-free maneuvers exist. This modules can be seen as a watchdog for an automatically driving vehicle or as part of a driving assistant system. It consists of four main steps: *occupancy prediction*, *trajectory planning*, *trajectory tracking*, and *collision checking.* First, a set of possible future trajectories of other traffic participants is predicted (see Fig. 2, ①). Second, the trajectory planning algorithm generates possible trajectories of the ego car (Fig. 2, ②); then it is checked if the generated trajectories do not intersect with the predicted occupancy of the other traffic participants (Fig. 2, ③). Finally, if a very limited set of safe trajectories exists, the best emergency maneuver is enabled using trajectory tracking (Fig. 2, ④).

The objective of this work is to create a framework for automatically computing the occupancy sets of other traffic participants on arbitrary road networks. A fast and robust algorithm for computing the occupancy prediction is required in order to meet the real-time system constraints. The proposed algorithm for computing the over-approximative occupancy takes as main inputs the surrounding traffic participants and the map of the road network. The occupancy prediction is computed for consecutive time intervals instead of points in order to ensure that no collision is missed between points in time. The main idea of the algorithm is the following: First, given the models of traffic participants as a dynamical system stored in a database, we propose two types of abstractions of them as described in Sec. 2. The reachable set of the first abstraction gives the lower and upper bound in the longitudinal direction and the reachable set of the second abstraction gives the left and right bound in the lateral direction, as shown in Fig. 3. Finally, the intersection of the longitudinal and lateral bounds determines the over-approximation of the vehicle occupancy.

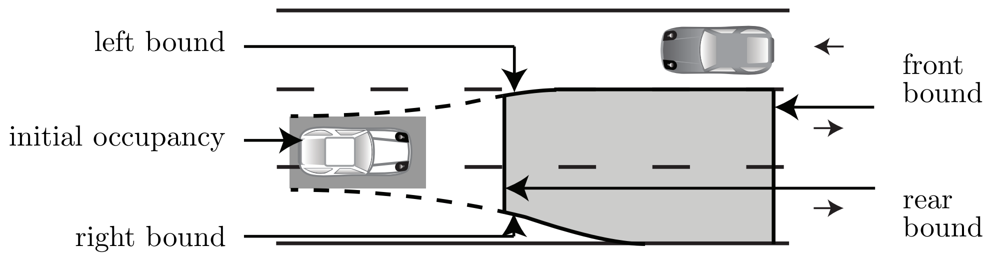


Figure 3: Initial occupancy and boundaries of the occupancy set for a long time interval.

The following assumptions are made throughout the paper: the data related to the vehicles placed in proximity of the ego car is known. Moreover, it is assumed that the required information, position, orientation, length, width, velocity, acceleration of each car and the map, is available in a given time interval, but these values can be also assumed as uncertain within bounded sets.

3.3 Mathematical Modeling

In this subsection we present the formal representation of the road network and the model of the vehicle that is used for computing the occupancy prediction.

3.3.1 Road Network Representation

A formal and robust representation of the road network is required in order to compute the occupancy prediction. To this end, *lanelets* [28] are used, which are atomic, interconnected, and drivable road segments.

Definition 3.1 (Lanelet) *A lanelet is defined by its left and right bound, where each bound is represented by an array of points (a polyline), as shown in Fig. 4.*

The geometry of the lane boundaries is either provided from a map or by algorithms which build the map online using onboard sensors, see e.g. [29], [30]. Without loss of generality we assume that the lane information is provided by a map from now on.

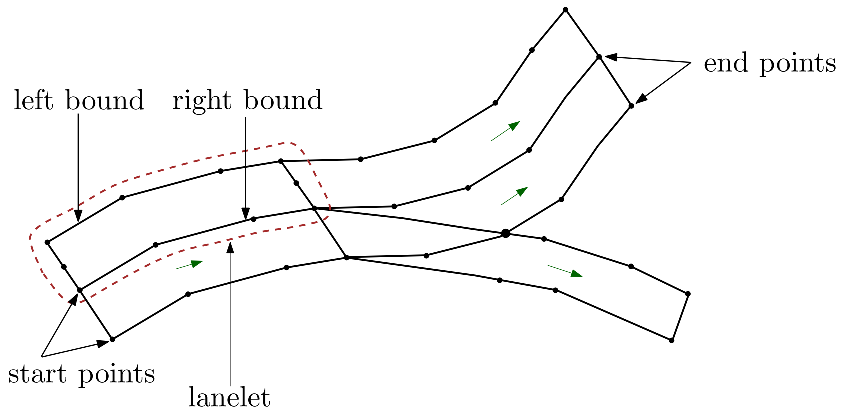


Figure 4: Lanelets

The lane boundary information for lanelets can be directly obtained from an open-source map (e.g. OpenSteetMap[1]) by importing a raw map into the free editor *JavaOpenStreetMap* (JOSM)[2]. We additionally annotate the raw data consisting of *left border* and *right border* by the *speed limit* (maximum allowed speed on that specific lanelet), a unique *ID*, the *lane width*, and the *driving direction*. Without loss of generality, it is assumed that all laterally adjacent lanes have the same length when creating the lanelets using JOSM. This is exemplarily shown for the road network in Fig. 4, where lanelet 1 has the same length as lanelet 2. This assumption is useful for representing the outgoing transition from one lane to another.

[1]`www.openstreetmap.org`

[2]`https://josm.openstreetmap.de`

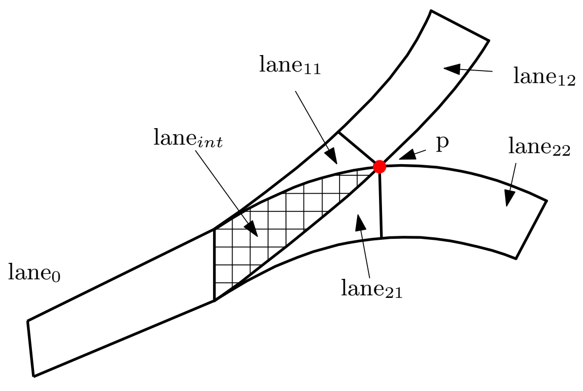
In order to obtain all future trajectories of a vehicle, an *adjacent lanelet matrix* is computed upfront. For this purpose, we represent the road network as a *directed graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the vertices $\mathcal{V}$ are represented by the lanelets and the directed edges $\mathcal{E}$ represent the possible transition between two adjacent lanelets, where each node has four types of outgoing edges: longitudinal, left, right, empty. The adjacent lanelet matrix $A_\mathcal{G}$ is defined as follows: $A_\mathcal{G} : \mathcal{V} \times \mathcal{V} \rightarrow \{long, right, left, \emptyset\}$, where $\times$ denotes the Cartesian product.

We define *start points* and *end points* of a lanelet as the first and the final points of the left and right border in driving direction, as shown in Fig. 4. Two lanelets are called *longitudinally adjacent*, i.e., $A_\mathcal{G}(\text{lanelet}_1, \text{lanelet}_2) = \text{long}$, if the left and right start point of one lanelet are identical with the corresponding final points of the other lanelet. We say that lanelet_2 is the left-adjacent lane of lanelet_1, i.e., $A_\mathcal{G}(\text{lanelet}_1, \text{lanelet}_2) = \text{left}$, if the points of the left border of lanelet_1 are identical to the ones of the right border of lanelet_2. This is analogously defined for right-adjacent lanes. For implementation reasons, one might accept small deviations of connection point of lanelets rather than demanding that the values are identical.

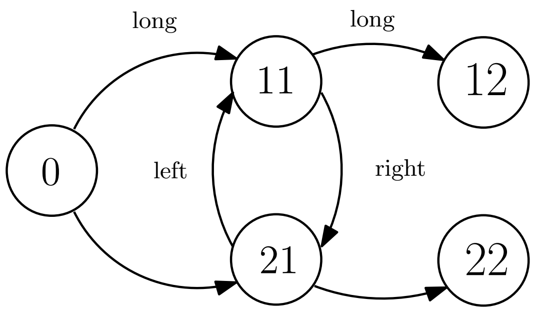
For road networks which contain only longitudinal and laterally adjacent lanes, the construction of the adjacent road matrix is straightforward. However, most road networks contain also road bifurcations, see Fig. 4, which have to be constructed in a special way to ensure that the computed occupancies are over-approximative. This is ensured by modeling possible lane changes as long as there exists an intersection of lanes as shown in Fig. 5a. Since we can only model that along the full length of a lanelet one can perform a lane change or one cannot perform a lane change at all, we have to partition the lanelets accordingly. Therefor, we introduce the point p as the intersection of the outer[3] lane bounds of the bifurcating lanes, see Fig. 5a. If the final points of the outer bounds of lanelet_{11} and lane_{21} correspond with the point p, and lanelet_{21} and lane_{22} continue the corresponding lanes as shown in Fig. 5a, all lanelets fulfill the constraint that they are either adjacent along their full length or not at all. The resulting adjacency matrix is presented in Fig. 5b. The adjacency makes it possible to define lanes:

Definition 3.2 (Lane) *A lane is defined as the union of lanelets, which are longitudinally adjacent.*

[3] In a left curve, the outer bound is the right border and the left bound in a right curve.



(a) Lanelets



(b) Adjacency graph

Figure 5: Road bifurcation.

3.3.2 Model of Other Traffic Participants

In this part, the mathematical model of the other traffic participants is derived. We use the approach from [31], where a simple model which satisfies the following constraints is considered:

$C1$: positive longitudinal acceleration is stopped when a parameterized speed $v_{\max}$ is reached ($v_{\max}$ could be set to a certain percentage above the official speed limit);

$C2$: driving backwards in a lane is not allowed;

$C3$: positive longitudinal acceleration is inversely proportional to speed above a parameterized speed v_S (modeling a maximum engine power);

$C4$: maximum absolute acceleration is limited by $a_{\max}$;

$C5$: actions that cause leaving the road/lane/sidewalk/crosswalk boundary are forbidden. Crossing lanes is allowed if not permitted by lane markings and traffic signs.

While $C3$ and $C4$ are physical constraints, the other constraints are derived from the traffic rules listed in Vienna Convention on Road Traffic [32]. The considered constraints can model the uncertain behavior of the other traffic participants. Of course, further constraints which reflect other traffic rules can be considered. However, removing some of the constraints does not affect the soundness of the verification procedure but it increases the behavior uncertainty of the other traffic participants, which leads to a more conservative behavior of the ego vehicle.

In the following we consider that the dynamics of the vehicle are modeled by a point mass (while a rectangle is considered to enclose their size), using the approach described in [33]:

$$\ddot{s}_x(t) = a_x(t), \quad \ddot{s}_y(t) = a_y(t), \tag{2}$$

where $s_x(t), s_y(t)$ describe the position and $a_x(t), a_y(t)$ describe the acceleration. In order to restrict $a_x(t)$ and $a_y(t)$ according to the constraints $C1$-$C4$, unit vectors that point towards the longitudinal and lateral direction of the vehicle are introduced: $\Phi(t)^{\text{long}} = \frac{1}{v}[v_x(t),\, v_y(t)]^T$, $\Phi(t)^{\text{lat}} = \frac{1}{v}[-v_y(t),\, v_x(t)]^T$, where $v = \|[v_x, v_y]^T\|_2$. Thus, we can define a_x, a_y as a function of the longitudinal acceleration $a^{\text{long}}(t)$ and the lateral acceleration $a^{\text{lat}}(t)$:

$$\begin{bmatrix} a_x \\ a_y \end{bmatrix} = \Phi^{\text{long}} a^{\text{long}} + \Phi^{\text{lat}} a^{\text{lat}}.$$

Let us define a normalized steering input u_1, where $u_1 = \pm 1$ represents steering to the left or right using the full tire friction potential; the lateral acceleration can be computed using the following equation:

$$a^{\text{lat}} = a_{\max} u_1.$$

In order to consider constraint $C4$, the remaining acceleration potential in longitudinal direction is limited to

$$a^{\text{long}}_{\text{c1}} = \sqrt{a^2_{\max} - {a^{\text{lat}}}^2}.$$

The maximum longitudinal acceleration for the engine power P and the vehicle mass m is $\frac{P}{m\,v} = a_{\max}\frac{v_S}{v}$, where $v_S = \frac{P}{a_{\max} m}$ is the speed above which the acceleration is limited by the engine power and no longer by the tire friction. However, since this parameter cannot be easily estimated, another option is to set $v_S = \infty$, which provides an over-approximation of the occupancy set. Similarly to the lateral acceleration, a normalized control input u_2 for the longitudinal acceleration is introduced, where $u_2 = \pm 1$ represents full braking and full acceleration within the acceleration potential. Limited engine power, the restriction to forward driving, and the maximum speed (constraints $C1$-$C3$) are considered by limiting the acceleration to

$$a_{\text{c2,long}} = \begin{cases} a_{\max}\frac{v_S}{v}, & v_S < v < v_{\max} \wedge \tilde{u}_2 > 0 \\ a_{\max}, & (0 < v \leq v_S \vee (v > v_S \wedge \tilde{u}_2 \leq 0)) \\ & \wedge v < v_{\max} \\ 0, & v \leq 0 \vee v \geq v_{\max} \end{cases}$$

Combining a_{c1}^{long} and a_{c2}^{long} results in the longitudinal acceleration which satisfies the proposed constraints $C1$-$C4$ ($C5$ for leaving the road is considered later):

$$a_{\text{long}} = \begin{cases} a_{\text{c2,long}}\,\tilde{u}_2, & |a_{\text{c2,long}}\,\tilde{u}_2| \le a_{\text{c1,long}} \\ a_{\text{c1,long}}\,\text{sgn}(\tilde{u}_2), & |a_{\text{c2,long}}\,\tilde{u}_2| > a_{\text{c1,long}}, \end{cases}$$

Of course, *static obstacles*, e.g. stationary vehicles or just other objects, can also be considered; they can be modeled as a particular case of a traffic participant, where the acceleration and velocity is zero.

3.4 Set Prediction

In verification of real-time systems, one of the most important constraints is computation time since the control input must be applied in a certain amount of time. In addition, for safety-critical systems, control actions should be verified before execution. Automated driving is a typical example of a safety-critical, real-time system and thus it has to address the aforementioned constraints. In the collision avoidance concept proposed in [33], one of the most time-consuming steps is the *occupancy prediction* of the other traffic participants. Hence, in order to guarantee safety, the occupancy prediction must be performed within a well-defined time interval.

In order to meet the safety requirements, we need a fast and reliable algorithm for occupancy prediction of other traffic participants. To this end, we use the results from Sec. 2 for the computation of the over-approximative occupancy. The more abstract models as defined in Sec. 2 are considered, the more accurate the occupancy is, but as a drawback, the computing time increases. So a trade-off between accuracy and computation time must be found.

Subsequently we consider two abstract models for computing the occupancy of each traffic participant. The first abstraction is denoted by M_1 and it considers only constraint $C4$. The predicted occupancy set computed using abstraction M_1 is denoted by m_1 and we call it occupancy towards road boundaries. The second abstraction denoted by M_2 considers the constraints $C1, C2, C3$. The occupancy set along road boundaries using model M_2 is denoted by m_2. Finally, the constraint $C5$ is considered and the intersection between m_1, m_2, and road network boundaries is performed. The intersection is denoted by *occ* and it represents the over-approximative occupancy of the vehicle real model.

Algorithm 1 provides the overall occupancy prediction for the other traffic participants. The occupancy prediction algorithm has the following inputs: the time horizon for which the prediction is done t_h, time step Δt and the vehicle information for each traffic participant

Algorithm 1 Over-approximative occupancy.

```
Input: $\Delta t$, $t_h$, $cars$, $lanes$, $road\_network$
Output: $occ$
1: parfor $car_i \in cars$
2: $occ_i \leftarrow \emptyset$
3: $occ_{m1} \leftarrow$ Occ_TowardsRoadBoundary($car_i, \Delta t, t_h$);
4: for all $lane_j \in lanes$
5: $occ_{m2} \leftarrow$ Occ_MinMaxAcceleration($car_i, \Delta t, t_h, lane_j$);
6: $occ \leftarrow$ Occ_StayOnTheRoad($occ_{m1}, occ_{m2}, lane_j$);
7: $occ_i \leftarrow merge(occ_i, occ)$;
8: end for
9: end parfor
```

car_i. Additionally, the following inputs are required: *lanes* and *road_network*, which are formalized in Sec. 3.3.1. As stated in the Sec. 3.2, in the current setting we assume that the map information is known. For each vehicle, first the occupancy prediction in lateral direction is computed with Occ_TowardsRoadBoundary. Then the occupancy in longitudinal direction is predicted, using Occ_MinMaxAcceleration. Finally, the intersection between the first and second occupancy prediction with the lane boundaries, computed with Occ_StayOnTheRoad returns the over-approximated occupancy prediction. Furthermore, in order to improve the performance, the algorithm is parallelized for every car. In the following subsections the occupancy prediction for each considered abstract model is presented.

3.4.1 Occupancy Towards the Road Boundary (Abstraction 1)

The first abstraction of the vehicle dynamics considers the limited absolute acceleration. Also, the reachable set of the proposed abstract model returns the occupancy of the vehicle towards road boundaries. This abstraction makes it possible that the occupancy of each car, at a specific moment in time t, can be described by a circle as mentioned in [21] with center c and a radius r (see Fig. 6):

$$c(t) = \begin{bmatrix} x(0) \\ y(0) \end{bmatrix} + \begin{bmatrix} v_x(0) \\ v_y(0) \end{bmatrix} t, r(t) = \frac{1}{2} a_{max} t^2, \tag{3}$$

where x, y represent the current position of the vehicle and v_x, v_y, a_{max} are the current velocity in x and y direction, respectively maximum acceleration of the vehicle. As mentioned before, we want to compute the occupancy for time intervals, as shown in Fig. 6, e.g. the occupancy

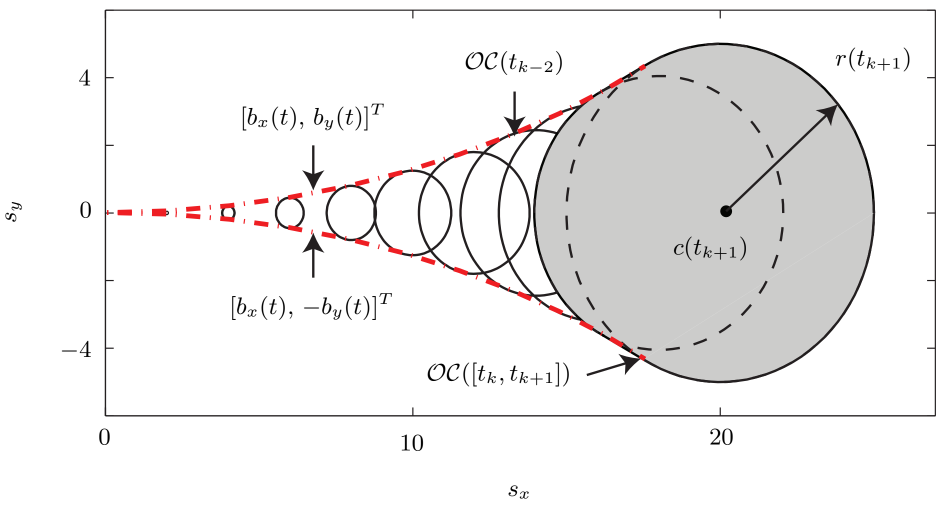


Figure 6: Occupancy sets.

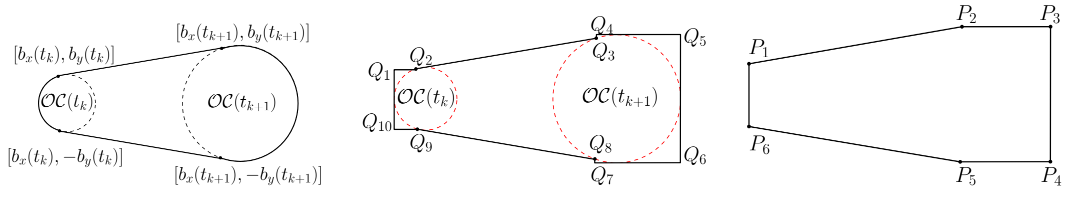
of the car between the moment t_k and t_{k+1}. In [33] it is proven that the boundary of the occupancy is given by the following equation, assuming that $x = 0, y = 0, v_x = v$ and $v_y = 0$:

$$b_x(t) = v_0 t - \frac{a_{max}^2 t^3}{2v_0}, \quad b_y(t) = \sqrt{\frac{1}{4}a_{max}^2 t^4 - (\frac{a_{max}^2 t^3}{2v_0})^2}. \tag{4}$$

In order to prevent driving backward, the solution in x-direction is to set the maximum value of $b_{x_{max}} = b_x(t_{max})$ for $t \geq t_{max}$. The value of t_{max} is found by solving the equation $b_x^{'}(t) = 0$.

$$v_0 - \frac{3a_{max}^2 t^2}{2v_0} = 0 \Rightarrow t_{max} = \sqrt{\frac{2}{3}\frac{v_0}{a_{max}}}. \tag{5}$$

Recall that our goal is to compute the occupancy for a given time interval $[t_k, t_{k+1}]$. Hence, the occupancy between t_k and t_{k+1} is given by the circles corresponding with the times, t_k and t_{k+1}, $\mathcal{OC}(t_k)$, $\mathcal{OC}(t_{k+1})$ and the boundary of the occupancy, as shown in equation 4, see Fig. 6.



(a) Boundary of occupancy **(b)** Enclosing polytope **(c)** Convex polytope

Figure 7: Occupancy set. Computation steps.

We propose a three-step method in order to predict a robust occupancy set towards a road boundary for a given time interval. First, we approximate the boundary of the occupancy with the *convex hull* of circles $\mathcal{OC}(t_k)$ and $\mathcal{OC}(t_{k+1})$, as shown in Fig. 7a. Let $\mathcal{X}$ be a finite set, with $x_i \in \mathcal{X}$ and $i > 0$. The convex hull of the set $\mathcal{X}$ is a set of all convex combination of x_i:

$$\mathrm{Conv}(\mathcal{X}) = \Big\{ \sum_{i=1}^{|\mathcal{X}|} \lambda_i x_i \mid \forall i : \lambda_i > 0, \sum_{i=1}^{|\mathcal{X}|} \lambda_i = 1 \Big\}.$$

Thus, the occupancy set for the time interval $[t_k, t_{k+1}]$ is enclosed by the circles $\mathcal{OC}(t_k)$, $\mathcal{OC}(t_{k+1})$ and the boundary determined by the vertices $[b_x(t_k), b_y(t_k)]$, $[b_x(t_{k+1}), b_y(t_{k+1})]$, $[b_x(t_k), -b_y(t_k)]$ and $[b_x(t_{k+1}), -b_y(t_{k+1})]$. For approximating a circle with a polygon, a large number of vertices is needed. Hence, the occupancy polygon is also determined by a large number of vertices. A *polygon* is determined by a finite number of coplanar line segments such that each segment intersects exactly two others, one at each of its endpoints.

Taking into account that the polygon intersection algorithm is performed online, and the time complexity depends on the number of vertices, we approximate each of the occupancy circles with an appropriate polygon, as shown in Fig. 7b. Each circle is enclosed by a rectangular shape, starting and ending with the vertices which determine the boundary. The resulting occupancy set is a non-convex polytope, determined by 10 vertices: $\{Q_1, Q_2, Q_3, Q_4, Q_5, Q_6, Q_7, Q_8, Q_9, Q_{10}\}$, where the vertices which determine the boundary of occupancy and which merge the two rectangle shape are: $Q_2 = [b_x(t_k), b_y(t_k)]$, $Q_3 = [b_x(t_{k+1}), b_y(t_{k+1})]$. $Q_8 = [b_x(t_{k+1}), -b_y(t_{k+1})]$, and $Q_9 = [b_x(t_k), -b_y(t_k)]$. The other vertices can be easily computed. Finally, we consider that the over-approximative occupancy set for the time interval $[t_k, t_{k+1}]$ is given by the convex hull of the enclosing polytope, as shown in Fig. 7c. This approach is considered as a trade-off between the high number of the vertices and the accuracy of the approximation. Thus, the occupancy of the point mass is a convex polytope given by the vertices $\{P_1, P_2, P_3, P_4, P_5, P_6\}$.

Let us denote with $c(t) = [c_x(t), c_y(t)]$ the center of the occupancy of a point mass at time $t \in \{t_k, t_{k+1}\}$. Then, one can determine the vertices of the over-approximative occupancy between time step t_k and t_{k+1} as follows, taking into account the aforementioned computation steps:

$P_1 = [c_x(t_k) - r(t_k), c_y(t_k) - b_y(t_k)]$;

$P_2 = [c_x(t_{k+1}) - b_x(t_{k+1}), c_y(t_{k+1}) + r(t_{k+1})]$;

$P_3 = [c_x(t_{k+1}) + r(t_{k+1}), c_y(t_{k+1} + r(t_{k+1}))]$;

$P_4 = [c_x(t_{k+1}) + r(t_{k+1}), c_y(t_{k+1} - (t_{k+1}))]$;

$P_5 = [c_x(t_{k+1}) - b_x(t_{k+1}), c_y(t_{k+1}) - r(t_{k+1})]$;

$P_6 = [c_x(t_k) - r(t_k), c_y(t_k) + b_y(t_k)]$.

Due to convexity and position invariance, it suffices to compute the occupancy prediction polygon of each vertex of the rectangle enclosing the vehicle Occ_{rl}, Occ_{fl}, $Occ_f r$ and Occ_{rr}.

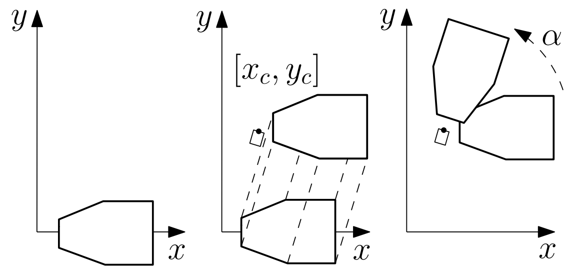


Figure 8: Geometrical transformation - occupancy.

For simplicity, it is assumed that the enclosing rectangle coincides with the four extremities of the vehicle, *rear-left*, *front-left*, *front-right*, and *rear-right* respectively. Each occupancy polygon is then *translated* and *rotated* according to the current tire position $[x_c, y_c]$ and orientation α as shown in Fig. 8.

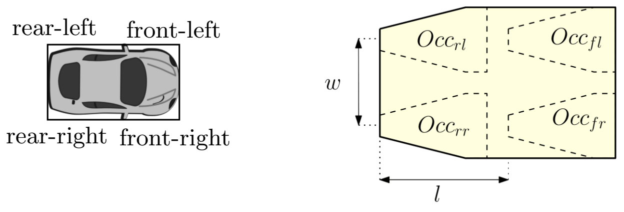


Figure 9: Occupancy prediction for a vehicle, for a specific time interval.

Finally, the over-approximated occupancy prediction of each vehicle is done by computing the convex hull between the vertices of each occupancy polygon, Occ_{rl}, Occ_{fl}, Occ_{fr}, Occ_{rr}, see Fig. 9.

3.4.2 Occupancy Along the Road Boundary (Abstraction 2)

First, the *merged lanes* are constructed by connecting two longitudinally adjacent lanes within the *vehicle route network graph.* In this deliverable, we assume that the front and rear bound is determined by following the center polyline of each lane as defined in Sec.3.3.1 (future work will consider the actual optimal solution). The algorithm used to compute the *lower bound* and *upper bound* of the occupancy is presented in Algorithm. 2.

The position corresponding to the rear bound x_{min} of the occupancy is computed by applying full deceleration to the minimum position of the car $x_{car_{min}}$, for the current velocity v, as described in [33]; then, the position corresponding to the front bound of the occupancy is computed likewise, by applying full acceleration to the $x_{car_{max}}$. The next step is to find the previous and the next points within the trajectory for x_{min}, q_i and q_{i+1}, as shown in Fig. 10. Then, the perpendicular line p_i on the segment determined by the previous and next points $[q_i, q_{i+1}]$ is constructed, such that the minimum point, x_{min} is contained by the perpendicular

Algorithm 2 Occupancy along road boundary.

```
Input: $t$, $car$, $lanes$, $road\_network$, $graph\_adj$, $traj$
Output: $occ\_along\_boundaries$
 1: $occ\_along\_boundaries = \emptyset$;
 2: $x_{min} \leftarrow x_{car_{min}} + vt - \frac{1}{2}at^2$;
 3: $x_{max} \leftarrow x_{car_{max}} + vt + \frac{1}{2}at^2$;
 4: find $q_i, q_{i+1}$ s.t. $q_i, q_{i+1} \in traj, q_i \leq x_{min} \leq q_{i+1}$;
 5: find $q_j, q_{j+1}$ s.t. $q_i j q_{j+1} \in traj, q_j \leq x_{min} \leq q_{j+1} l$
 6: find line $p_i \perp [q_i, q_{i+1}]$ s.t. $x_{min} \in p_i$;
 7: find line $p_j \perp [q_j, q_{j+1}]$ s.t. $x_{max} \in p_j$;
 8: $occ\_along\_boundaries = p_i \cap p_j \cap lane$;
 9: if left adjacency $\neq \emptyset$ then
10:     repeat $4 - 8$ for $traj_{left}$;
11: end if
12: if right adjacency $\neq \emptyset$ then
13:     repeat $4 - 8$ for $traj_{right}$;
14: end if
```

line $x_{min} \in p_i$. The perpendicular line represents the rear bound of the occupancy, see the red line in Fig. 10. If the current lane has lateral adjacency, then the rear boundary is intersected with the trajectory within the lateral lane; the output is point x'_{min}. Finally, the steps 4-8 are repeated for the new x'_{min} and the trajectory if the lateral lane. The rear bound of the laterally adjacent lane is the blue one as sown in Fig. 10. The computation of the front boundary is analogous.

Finally, the intersection of the rear and front bound with the road boundaries gives the occupancy for the considered abstract model.

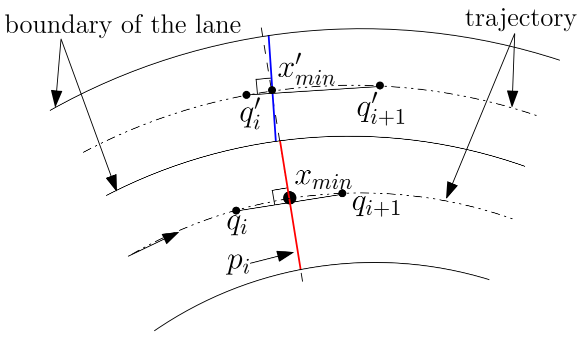


Figure 10: Boundary for lateral adjacency.

3.4.3 Intersection between reachable sets of the abstract models

The last step in computing the over-approximative occupancy prediction is to intersect the occupancy of all abstract models with the road boundary. Since the second abstraction already considers the intersection with the road boundaries, the over-approximative occupancy is given by the intersection between the occupancy of the first and the second abstraction.

Since the intersection between occupancy of M_1 and occupancy of M_2 is done in real time, a fast and efficient clipping algorithm is needed [34]. Taking into account these requirements, the most fitting algorithm is *Greiner Hormann Polygon Clipping Algorithm* [35]. The main advantage of this algorithm is that it can deal with non-convex shapes; in the current setting, the occupancy of M_2 is non-convex because of the road boundary shape. The algorithm takes as an input two polygons: the *subject polygon* S is the polygon which is being cut and the *clipping polygon* C, as shown in Fig. 11.

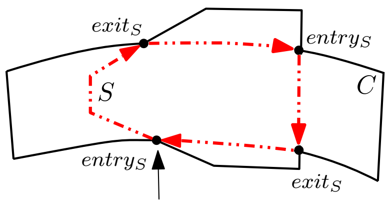


Figure 11: Intersection between the occupancy of M_1 and the occupancy of M_2.

C is the polygon against which S is being cut. In this case, the subject polygon is the occupancy of M_1 model and the clipping polygon is the occupancy of the M_2 model. The occupancy of M_2 is the clip polygon because it is already intersected with the road boundaries. Thus, the predicted occupancy is clipped against the road boundaries, prohibiting the vehicle leaving the road.

3.5 Numerical Experiments

3.5.1 Simulation Results

In the following subsections three different lane scenarios taken from OpenStreetMap [36] and processed with *JOSM* [37] are presented.

Scenario 1. In the first scenario, a road bifurcation is considered, as shown in Fig. 12. The occupancy prediction is computed starting with $t_0 = 0$, for a time horizon $t_h = 3$ seconds and the step time for computing the occupancy interval is $\Delta t = 0.5$.

The information about the vehicle, initial position $[x, y]$, orientation α, maximum acceleration a_{max}, maximum velocity v_{max}, the speed above which the acceleration is limited by

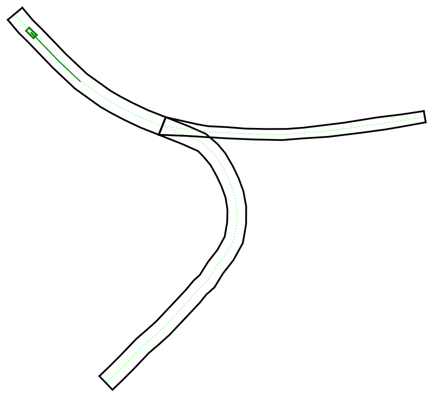


Figure 12: Scenario 1. Map representation.

Table 1: Scenario 1: Vehicle parameters.

α	v	a_{max}	v_{max}	v_{sw}	w	l
-0.778	29.9	5	50	30	1.8	4.2

the engine power v_{sw}, length l and width w, are shown in Table 1. In this situation there are two possible trajectories which the vehicle can follow as shown in Fig. 13.

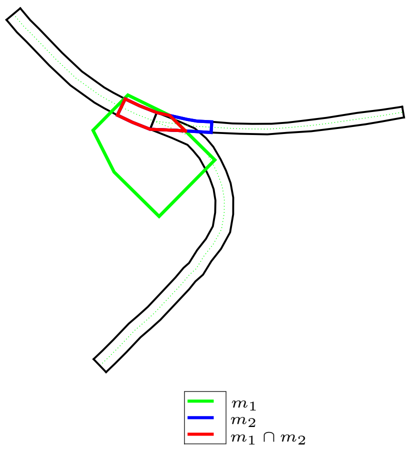
Scenario 2. The vehicle information is in Tab. 2 and the occupancy prediction is shown in Fig. 14. In this situation, a lane change is possible towards both directions *left* and *right*.

Table 2: Scenario 2. Vehicle parameters.

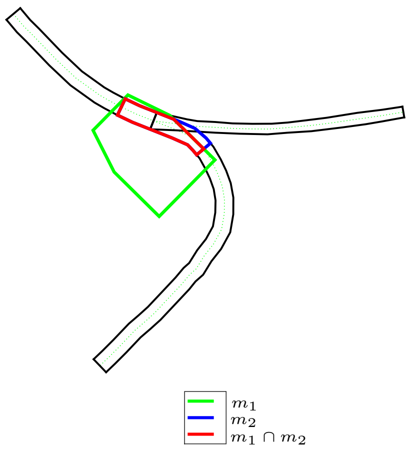
α	v	a_{max}	v_{max}	v_{sw}	w	l
0.014	19.9	5	50	30	1.8	4.2

Scenario 3. The vehicle information are presented in Tab. 3 and the occupancy prediction is shown in Fig. 15. In this scenario, the vehicle can perform a lane change towards the right side.

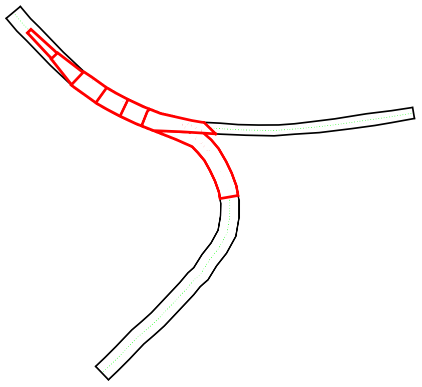
The online occupancy prediction framework is implemented in MATLAB. The computation times on a laptop (Intel Core i7 Processor with 2,2 GHz and 16 GB memory) are presented in Tab. 4 for each considered scenario.



(a) Occupancy set. Trajectory 1. $t_0 = 2$.



(b) Occupancy set. Trajectory 2. $t_0 = 2$.



(c) Occupancy prediction. $t_h = 3$.

Figure 13: Scenario 1: Occupancy prediction.

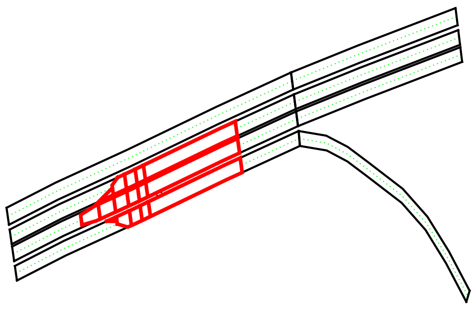


Figure 14: Scenario 2. Occupancy prediction.

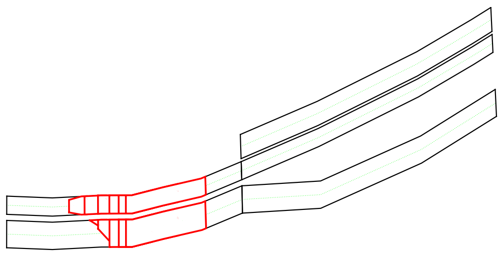


Figure 15: Scenario 3. Occupancy prediction.

Table 3: Scenario 3. Vehicle parameters.

α	v	a_{max}	v_{max}	v_{sw}	w	l
-0.382	19.9	5	50	30	1.8	4.2

3.6 Conclusion on Road Traffic Prediction

A framework for automatically computing the occupancy prediction of other traffic participants is presented. The proposed algorithm guarantees safety by considering all possible trajectories and is able to handle arbitrary road networks. The performed simulations show that our algorithm is fast enough for a real-time application. For the considered scenarios, the time for computing the over-approximative prediction sets for each vehicle is less than 0.2 seconds, for a 3 second prediction horizon. Our modular framework can be easily integrated in a collision avoidance system. Future work includes considering interaction between vehicles in occupancy prediction algorithm and also including a wider range of traffic rules.

4 Set-Based Prediction of Humans for Human-Robot Coexistence and Collaboration

Predicting the occupancy of a human in real-time is of great interest for the application *human-robot interaction* in UnCoVerCPS, in order to compute unsafe regions in human-avoidance strategies. The human body is composed of joints and links, suiting approximation by a kinematic chain, but the control strategy of the human is completely unknown. Consequently, the potential occupancy grows very fast compared to the one of the robot and it is difficult to compute a tight bound on occupancy in real-time. As such, current models consider only known or probable movements and usually do not account for a range of human dimensions. Focusing on the human arm, we present a new method where motion-capture

Table 4: Time for computing occupancy prediction.

	Scenario 1	Scenario 2	Scenario 3
Time (sec)	0.19	0.18	0.17

data from infrared markers is fitted to two abstractions. Reachable sets of the abstraction, calculated in real time, are formally shown to enclose all possible future occupancies of the arm for different body dimensions. The abstraction is built from measurements of 38 human subjects and tested on movements from these subjects, and found to enclose the motion of the arm. Such an over-approximative representation can guarantee safety in collision avoidance algorithms.

4.1 Previous Work

To address the problem of unforeseeability, much work has been carried out on determining what natural human movement is. The work in [38] aims at determining the cost function, humans try to minimize. Morasso et al. [39] observe straight spatial hand trajectories in point-to-point movements and deduce that the human control system operates in Cartesian space rather than joint space. Flash and Hogan [40] present the minimum spatial jerk optimization criterion which fits well to observed motion data. Available muscular effort is also a factor in human movement; in [41], the authors present a muscle-effort minimization criterion which predicts the natural movement of humans holding weights relatively accurately, whereas in whole body movement, additional factors such as keeping balance may influence movement [42]. Finally, a combination of models [43] taking into account spatial and joint movement may result in a better fit than one particular model.

In our situation, however, we also do not know whether a human would behave “naturally” in an environment with obstacles and a robot. The presence of a robot is known to affect the movement of humans in the vicinity to varying degree [44]. Markov chains are used to model human dynamics in [45], and probabilistic reachability analysis based on the dynamics obtained is used in a collision avoidance algorithm. Ding et al. [46] present another set-based prediction of the human occupancy using a Hidden Markov Model (HMM). The authors point out that unforeseen or unusual movements would not be accounted for by their approach, and propose reachability analysis as a complementary technique to account for such movements. Assuming only a maximum obstacle speed, Vatcha and Xiao [47] present an algorithm using dynamic envelopes, which works for slower-moving obstacles. Other approaches to human avoidance forego a human model entirely in favor of a measure of safety based on robot state and distance from the human, which can be used to generate a reactive force to guide the trajectory of the robot away from the human [48] or slow the intended trajectory [49] [50]. While working well for normal interaction and in trials, these methods do not formally guarantee avoiding injury to the human.

Concerning the geometry of the human, Digital Human Models (DHMs) where both the skeleton geometry and the muscle placement are taken into account, such as the model presented by Holzbaur et Al. [51] and AnyBody [52], are well studied and provide insight into how humans move. Demircan et al. were successful in scaling and fitting the former model to data captured from a human using a Vicon system in real-time and achieved position errors of a few centimeters [53]. In this case, the geometry of the human under observation were known, and it was not intended to formally bound the position error. They also considered the production of maximum accelerations in the Cartesian space, showing how professional athletes controlled their muscles optimally to produce the desired maximum acceleration [54].

In contrast to these approaches to predicting human motion from sensor data, our goal is not simply to reduce the difference between the predicted motion and the observed motion, but to bound the possible occupancy of the human and to calculate this possible occupancy with real-time constraints. Our model should be as accurate as possible but must be over-approximative, so that the occupancy given by the model for a certain time interval includes all possible reachable occupancies of all shapes, sizes and masses of human arms to ensure that robots can safely avoid all humans. As such, DHMs have two drawbacks: Firstly, they do not account well for variation in size and mass parameters. Secondly, they are too complex to calculate in real time for all positions with set-based arithmetic.

Reachability Analysis [55] has been widely used to guarantee the safety of hybrid systems (i.e. systems with mixed discrete and continuous dynamics). Most approaches for reachability analysis can be categorized into simulation-based techniques (see e.g. [56]), Eulerian techniques (see e.g. [57]), and Lagrangian techniques. Simulation-based and Eulerian techniques suffer from an exponential complexity in the number of continuous state variables, limiting the applicability to rather low-dimensional systems. Lagrangian schemes propagate the reachable set for consecutive points in time or time intervals. Especially for linear continuous dynamics, large state spaces with potentially more than 100 continuous state variables can be efficiently computed. Typical set representations are: polytopes [58], zonotopes [59], ellipsoids [60], support functions [61], and oriented hyper-rectangles [62]. One challenge of this approach is that the set-based calculations required are rather time-consuming and for systems with nonlinear dynamics and high dimensionality, it can be very difficult to calculate occupancies online and within real-time.

Our proposed approach fits sensor data of the human arm to a simple mathematical model. In advance, we collect the set of parameters obtained from fitting motion capture data to the mathematical model to abstract it using differential inclusions, so that in real time we can perform inverse kinematics on sensor data, apply reachability analysis on the differential inclusions obtaining a reachable set of joint angles at a future point in time; and then transform this into an over-approximative potential occupancy at this time in the workspace. The occupancy can then be forwarded to a robot trajectory planner to avoid collisions with humans.

4.2 Problem Statement and Approach

The occupancy of the human body is fast changing and hard to predict: bodies have different dimensions, shapes and a large number of degrees of freedom. We focus on the movement of the human arm. Our approach has two phases: offline, we fit motion capture data to a kinematic model of the arm (Sec. 4.4), to determine parameters such as length, maximum accelerations and maximum joint velocities for a dynamic model (Sec. 4.5). Online and in real time, we fit sensor data to the kinematic model via inverse kinematics and use the dynamics obtained offline to predict the set of possible states of the model at a future time. Finally, we compute the potential occupancy in space of the arm as a mapping from that set of states. We test our human arm model on as yet unverified movements of the test subjects to validate the accuracy and the conservativeness of our occupancy prediction. We formalize the problem as follows.

Let $y(t)$ be the state of the human arm in its own, unknown state space $\mathcal{Y}$ as a function of time, and let $\dot{y} \in \gamma(y)$ be its uncertain dynamics, again unknown. Let $\mathcal{A}(y) \subseteq \mathbb{R}^3$ be the subset of space occupied by the arm in state y and $\mathcal{R}_y(t)$ be the reachable set (see (1)) of the arm.

Definition 4.1 (Exact Reachable Occupancy) *The exact reachable occupancy at a time $t = r$ is:*

$$^e\Gamma(r) = \{\mathcal{A}(y) | y \in \mathcal{R}_y(t)\}$$

As we can only obtain knowledge of the state of the robot from sensors, we define a sensor reading $s \in \mathcal{S} \in \mathbb{R}^n$, where $\mathcal{S}$ is the sensor space, i.e. the space of all possible sensor readings. We fit s to a simplified kinematic model with joint positions $q \in \mathcal{Q} \subseteq \mathbb{R}^n$ and velocities $\dot{q} \in \xi \subseteq \mathbb{R}^n$ we define the state of the kinematic chain $x = [q^\top, \dot{q}^\top]^\top$ and state space $\mathcal{X} = \mathcal{Q} \times \xi$. We define further the inverse kinematic map and the occupancy map:

$$map_{IK}(s) : \mathcal{S} \to \mathcal{P}(\mathcal{X}),$$
$$map_{OCC}(x) : \mathcal{X} \to \mathcal{P}(\mathbb{R}^3),$$

where $\mathcal{P}(\mathbb{R}^3), \mathcal{P}(X)$ are the power sets of $\mathbb{R}^3, \mathcal{X}$ respectively. The former mapping gives the set of states of the model possibly represented by some sensor data s, whereas the latter mapping gives the set of points in $\mathbb{R}^3$ possibly occupied by any human arm for a given state x of the model, accounting for variations in arm length and shape. Offline, we obtain the dynamics $\dot{x}(t) \in f(x(t))$ from analysis of data so that online we can compute the reachable set of x, $\mathcal{R}_x(t)$ from $\mathcal{R}(0) = map_{IK}(s)$, and define the reachable occupancy:

Definition 4.2 (Reachable Occupancy) *The reachable occupancy at a time $t = r$ is:*

$$\Gamma(r) \supseteq \{map_{OCC}(x) | x \in \mathcal{R}_x(r)\}$$

Our task is

- to find $\Gamma(r) \supseteq {}^{e}\Gamma(r)$,
- to compute $\Gamma(r)$ from s in a time faster than r,
- to keep $\Gamma(r)$ as small as possible.

We work with t in the range of 10-30ms, because after this $\Gamma(t)$ grows to cover the entire workspace. This is intended to interface with a safety controller as proposed in UnCoVerCPS and applied by the authors to human-robot interaction in UnCoVerCPS. We now show how the data was collected.

4.3 Data Collection

The prediction bases its validity on collecting extreme movement data from subjects of both genders with a wide range of ages and physical activity levels; this data is used for obtaining the uncertain model in advance.

Motion capture data was collected from 38 persons, 12 female and 26 male, ranging in age between 18 and 49 with a median age of 24. 50% did 3 or more hours of sport a week. The movements captured were designed to optimally cover the entire workspace of the human arm. The subjects were required to perform the following movements as fast as possible:

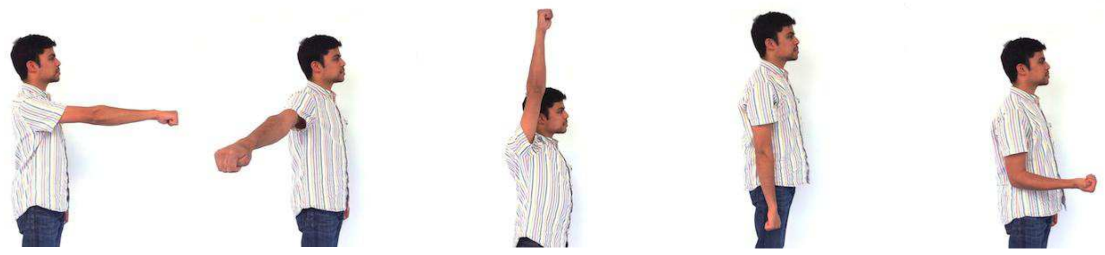


Figure 16: From left to right: Positions A, B, C, D and E.

- Punch to the front then recover to position E.
- Punch to the front, ending in position A.
- Position A to position B, elbow allowed to bend.
- Position A to position B, elbow not allowed to bend.
- Position C to position D, via position A.
- Position C to position D, via position B.

We capture the data using a 6-Camera Vicon Motion Capture system at a rate of 120Hz and filtered it using a 4$^{\text{th}}$ order Butterworth filter as in [53] to remove noise. The choice of filter was validated by the fact that the elbow joint accelerations agreed well with those measured by [63] using a camera. A higher order filter would have attenuated the joint accelerations whereas a lower order filter was found not to remove noise sufficiently. Inverse kinematics was then applied to the data sets to find the joint positions at each time step. The data is then fitted to a kinematic model in order to obtain parameters for this model, such as link length, maximum joint speeds and maximum accelerations.

4.4 Kinematic Model

In this section we present a familiar 4-degree-of-freedom (4-DOF) kinematic model of the human arm, which we then simplify to a 3-degree-of-freedom (3-DOF) model to reduce dimensionality and therefore computation time. We begin at the shoulder complex, which is composed of four joints, illustrated in Fig. 17. Of these, the most important is the ball-and-socket Gleno-humeral (GH) joint. The others have a limited range of motion and, in all cases, cause movement of the origin and orientation of the GH joint.

From the GH joint to the hand there are seven degrees of freedom, as shown in Fig. 18. The GH joint has three degrees of freedom and can be modeled by the 3 revolute joints 1–3. The elbow joint has two degrees of freedom: flexion of the forearm (radius and ulna, joint 4)

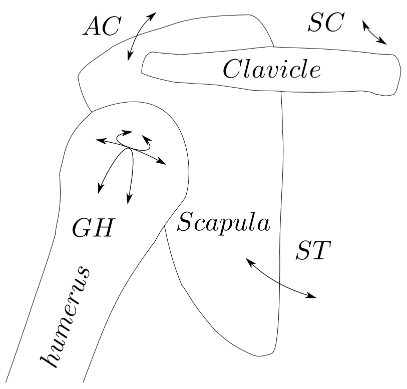


Figure 17: The shoulder complex. SC: Sternoclavicular joint, AC: Acromioclavicular joint, ST: Scapulothoracic joint, GH: Glenohumeral joint. All but the latter have limited motion; we focus attention on the GH joint, which is a ball-and-socket joint.

and rotation of the humerus along its own axis (joint 5). The latter does not greatly affect the occupancy of the forearm and hand, and neither do the joints in the wrist (joints 6 and 7) and the fingers. The occupancy resulting from movement of joints 5-7 through their entire range can be over-approximated by a static set. Therefore, there are only four degrees of freedom which determine occupancy of the arm.

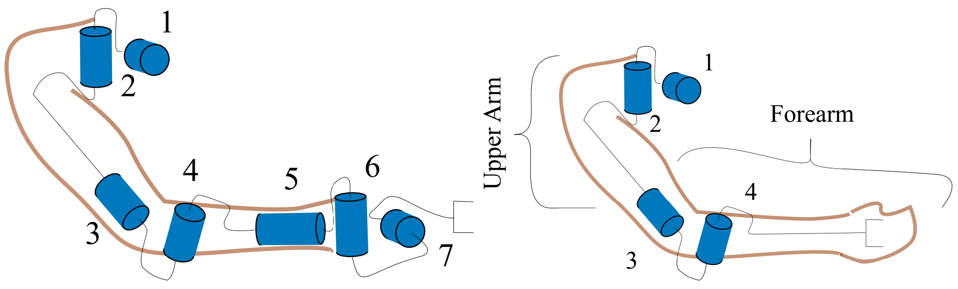


Figure 18: Left: seven degree of freedom model of human arm. Right: simplified four degree of freedom model of human arm excluding wrist and hand movement.

4.4.1 4-Degree-of-Freedom Model

A simplified kinematic model of the human arm is presented in [64] to calculate the workspace of the human arm. A similar model, identical except for the initial orientation of coordinate system is presented in [41]. In both cases, four degrees of freedom are used; the models differ only in terms of the initial orientation of the first joint.

In our model, the base coordinate system is defined by the four markers RSHO, CLAV, C7 and LSHO, as shown in Fig 19, and we used RELB and the midpoint of RWRA and RWRB for the inverse kinematics. We choose the base coordinate system such that the first joint corresponds approximately to adduction, the second to flexion, the third to rotation and

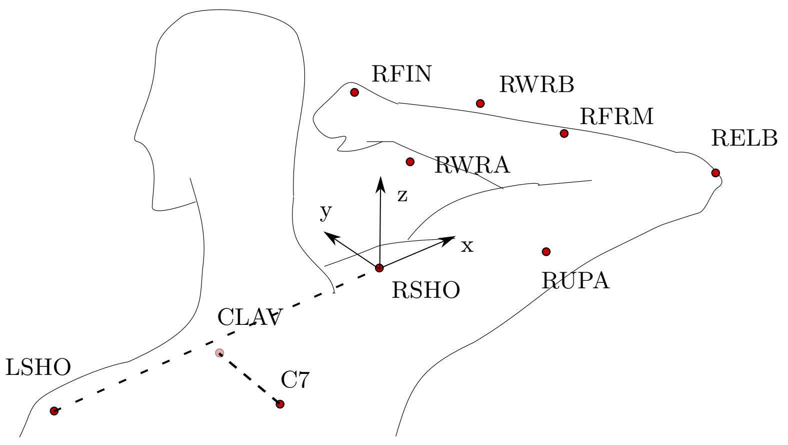


Figure 19: Markers and local base coordinate system. The marker CLAV is positioned at the base of the sternum between the two clavicles on the anterior of the torso. The markers CLAV, C7, LSHO and RSHO define the base coordinate system of the shoulder.

the fourth to elbow flexion. However, if the elbow approaches or passes close to the axis of joint 1 during a movement, this is a singularity in the workspace and the inverse kinematics registers large changes of the angle of joint 1 in the abstraction over a short time (see Fig. 20); the elbow movement is not redundant, as there are 2 rotational degrees of freedom and a two-dimensional workspace (the surface of a sphere with radius the length of the link), therefore the inverse kinematics yields an unique solution. As will be shown in Sec. 4.5, we wish to tightly bound the possible range of accelerations in the joint space, therefore we would like to ensure that the elbow cannot pass close to the axis of the first joint. As the axis of joint 1 can be freely chosen, we set it such that it does not intersect the workspace, as shown in Fig. 21. Therefore the singularity will not be reached in any movement of the arm.

By setting the origin at the shoulder marker, the shoulder rhythm, i.e. the movement and interdependencies of the other joints in the shoulder complex, as described in [65] is not considered as part of the arm movement.

We call the section of the kinematic chain from the shoulder to the elbow the *upper arm* and the section of kinematic chain from the elbow to the hand the *forearm*, as shown in Fig. 18. The position of the upper arm in the base coordinate system K_0 is defined by joints q_1, q_2 and the length of the first link. The position of the forearm is defined by the joints q_1, q_2, q_3 and q_4, and the length of the first and second links, although the position of the forearm relative to the coordinate system at the end of the upper arm, K_1 is only defined by q_3, q_4, and the length of the second link. As such, the entire kinematic chain can be considered to be two kinematic chains of two rotational joints followed by a link, joined together, identical except for the length of the links. This decomposition into two sub-chains

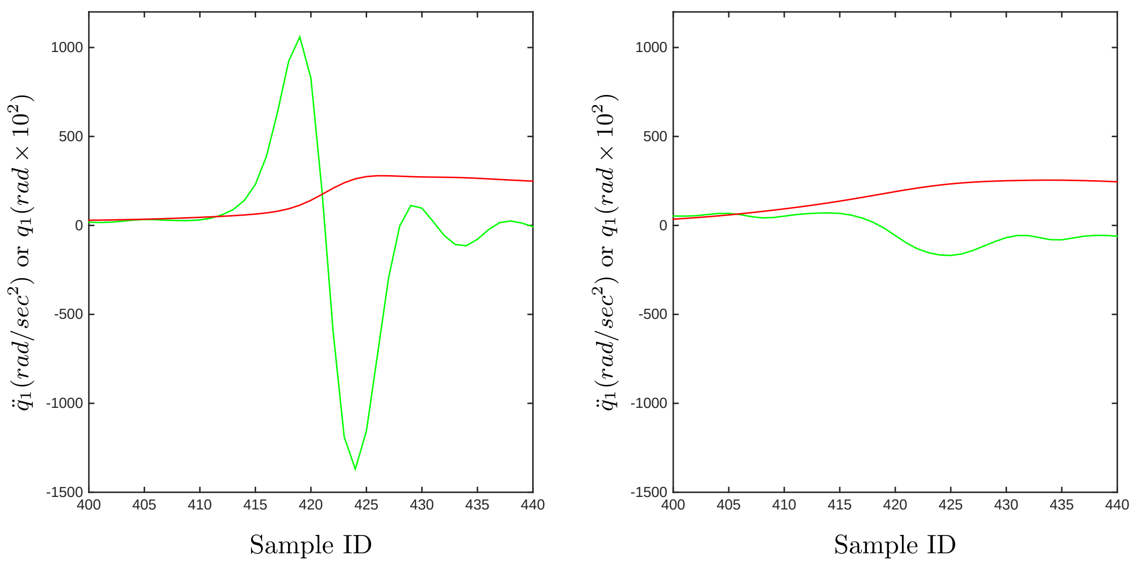


Figure 20: The effect of the orientation of the joint axis on the joint accelerations. The x-axis is the data sample and the y-axis is simultaneously the first joint position in $rad \times 10^2$ (red trace) and acceleration in rad/sec^2 (green trace). Left: axis is oriented as in Fig. 21a, and the movement of the elbow passes very close to the joint axis, resulting in high accelerations. Right: axis is oriented as in Fig. 21b; lower accelerations are seen.

is useful for computation of the potential occupancies, as discussed later in Sec. 4.6.

4.4.2 3-Degree-of-Freedom Model

The 4-DOF model can be simplified further, to reduce complexity and therefore computation time. The relationship between the 3rd and 4th joints is such that when the elbow is outstretched or bent in on itself ($|q_4| \approx \frac{\pi}{2}$), extremely large accelerations can be observed in rotation ($\ddot{q}_3$ is large), which is not the case when the elbow is at right angles ($q_4 \approx 0$), see Fig. 22. This is to be expected as in the latter case, the moment of inertia around the rotation axis (i.e. around the upper arm) is high, and in the former case it is negligible.

However, high accelerations and high velocities when $|q_4| \approx \frac{\pi}{2}$ do not lead to large changes in occupancy; the change in occupancy due to the third joint is therefore highly dependent on q_4. For this reason, it makes sense to merge the third and fourth joints into a single, prismatic joint: the arm is enclosed in a cylinder defined by the elbow and the hand, which has two rotational degrees of freedom around the shoulder. This is shown in Fig. 23.

Proposition 4.1 *The length and radius of the cylinder depend on the elbow flexion angle q_4 and the forearm and upper arm lengths l_f and l_h, thus:*

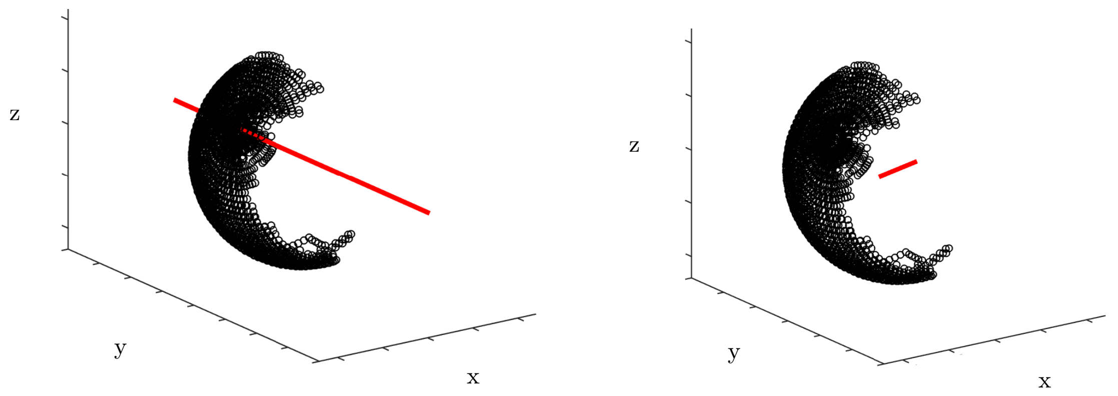


Figure 21: The workspace of the elbow, i.e. the possible positions of the elbow according to our data, and the axis of joint 1 in red. On the left, (a) the axis intersects the workspace, which could cause high $\ddot{q}_1$ when the upper arm aligns with the axis. On the right, (b), we rotate the axis so that it does not intersect the workspace and thus avoid high joint accelerations.

$$r = \frac{l_f l_h sin(q_4)}{\sqrt{l_f^2 + 4l_h^2 - 4l_f l_h cos(q_4)}} \tag{6}$$

$$l = \begin{cases} \sqrt{l_h^2 + l_f^2 - 2l_h^2 l_f cos(q_4) - r^2} & q_4 > q_{4a} \\ \sqrt{l_h^2 - r^2} & q_4 \leq q_{4a} \end{cases} \tag{7}$$

q_{4a} is the angle at which the maximum radius and the minimum length occur, and is calculated so:

$$q_{4a} = tan^{-1}(\frac{l_f}{2l_h}) \tag{8}$$

The maximum radius is $\frac{1}{2}l_f$.

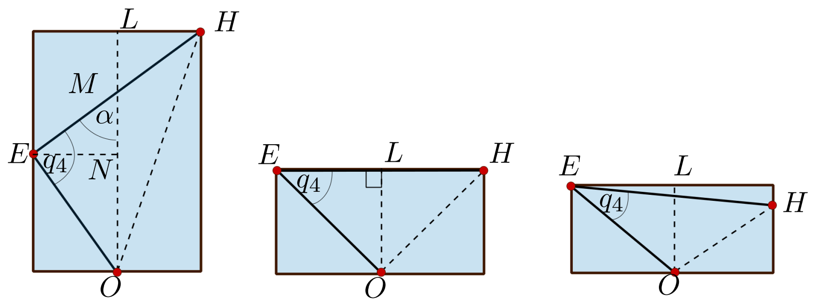


Figure 24

Proof 2 *In Fig. 24, O, E and H are the positions of the shoulder, elbow and hand respectively; the figure shows the section of the cylinder along the plane OEH. Let N be the base of the perpendicular from E onto OL. We wish to determine the cylinder length |OL| and*

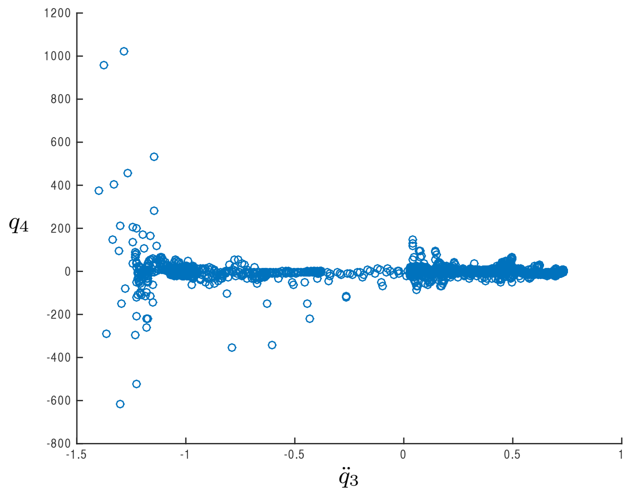


Figure 22: $\ddot{q}_3$ plotted against q_4 during one movement. The range at $q_4 = 0$ and that at $q_4 = -\frac{\pi}{2}$ are very different, but the large accelerations and velocities at the latter do not contribute to great changes in occupancy.

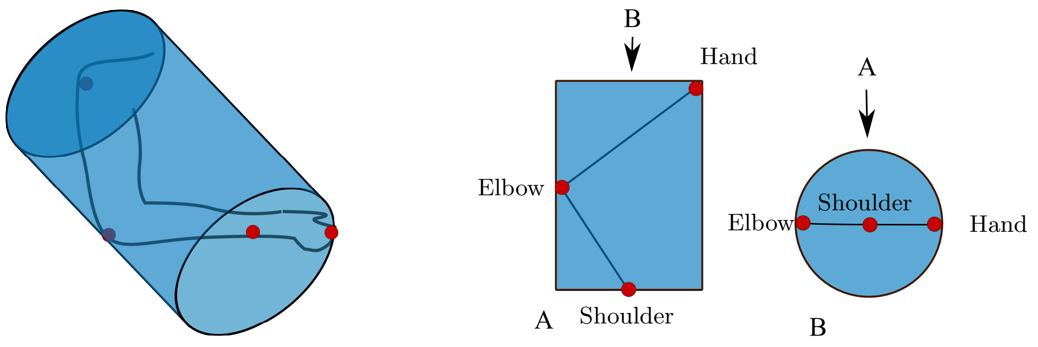


Figure 23: Left: Enclosing the arm with a cylinder. The hand lies on the intersection of the base of the cylinder and the side. The elbow lies on the side. The elbow joint angle determines length and radius of the cylinder, though for simplification, we always enclose the arm in the maximum radius cylinder.

radius $|EN|$. Let M be the intersection of OL and EH; M is then the midpoint of EH as LH and EN are both radii therefore HLM and ENM are similar. Knowing $|OE| = l_H$, $|EM| = \frac{l_F}{2}$ and q_4 we obtain $|OM|$ by the cosine rule:

$$|OM| = \sqrt{l_H^2 + \frac{l_F^2}{4} - l_H l_F cos(q_4)}$$

From the sine rule, we see that:

$$sin(\alpha) = \frac{sin(q_4)}{|OM|}|EM|$$

We now obtain $|EN| = |OE|sin(\alpha) = r$; substituting we find Eq. 6. We see that this is a *maximum in the middle diagram of Fig. 24, where $\alpha > \frac{\pi}{2}$; here $q_4 = tan^{-1}(\frac{l_f}{2l_h})$ as in Eq. 8*

In the case that $\alpha < \frac{\pi}{2}$, as in the left diagram of Fig. 24, the length is obtained from the right angled triangle OLH. $|OH|$ is obtained from the cosine rule on triangle OEH, yielding:

$$|OH| = \sqrt{l_H^2 + l_F^2 - 2l_H l_F cos(q_4)}$$

Pythagoras' Theorem then yields $|OL|$ as in the top part of Eq. 7. In the case that $\alpha > \frac{\pi}{2}$, as in the right diagram of Fig. 24, the triangle OEM determines the length, from which follows the bottom part of Eq. 7.

The simplest option in terms of representation is to enclose cylinders of smaller radius in the maximum. Therefore, the forearm movement is reduced to movement of a prismatic joint at the end of a kinematic chain as in Fig. 28.

4.5 Dynamic Model

As mentioned in Sec. 4.2, the arm is abstracted to a kinematic chain. The *state x* of a kinematic chain is $[\mathbf{q}^\top, \dot{\mathbf{q}}^\top]^\top$, where $\mathbf{q}$ is the vector of joint positions and $\dot{\mathbf{q}}$ is the vector of joint velocities. We discuss two models for determining the dynamics of the kinematic chain in Sec. 4.4 from the motion data gathered as described in Sec. 4.3 and therefore the reachable sets of the model – one model is torque-based and the other is acceleration-based. Finally, we motivate the use of the acceleration-based model.

4.5.1 Torque Model

Holzbaur et al. [51] collate the maximum muscle forces from a variety of studies for use in their DHM, thus it would be intuitive to calculate the range of torques of each joint as the sum of the range of contributions from each muscle, if the moment arms were known. Alternatively, Otis et al. [66] measure the maximum flexion, abduction and rotation torques for a range of shoulder joint positions and angular velocities, and Amis et al. [63] study the forces on the arm during maximal elbow movements, which is even more directly applicable to our model. Besides the literature mentioned, there is ample existing research into the maximal torques in the human arm on which to base a model. The kinematics of a serial-link manipulator can be described as:

$$\tau = H(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + F(\dot{\mathbf{q}}) + g(\mathbf{q}, x) + \tau_{ext}$$

where $\mathbf{q}$ is the vector of joint positions, H is the inertia matrix, C is the matrix of Coriolis and centrifugal forces, F is the friction vector and g is the gravitational vector as a function of the joint positions and orientation x. τ is vector of the total torque on the joints and τ_{ext} is the vector of external torques, including those from the movement of the base relative to the world. Calculating the reachable set of such a system presents several challenges:

1. Non-constant gravity vector – the base coordinate system of the arm is not fixed in space, it is attached to the shoulder and its orientation moves with that of the shoulder, as shown in Fig. 19. As such, the direction of gravity is not constant with respect to the base coordinate frame of the arm.

2. Presence of unknown external torques induced by the movement of the shoulder relative to the world.

3. Unknown mass and inertia parameters of the arm – these can be accounted for, but current methods e.g. [67] are calculation-heavy and introduce additional uncertainties in the over-approximation, thus would not be suitable for real-time calculation.

4. The system is highly nonlinear and would require linearization at every time step; this also leads to larger over-approximation and computation times that are not real-time-capable with current methods [67,68].

4.5.2 Acceleration Model

We present a model where acceleration is a linear function of the state, i.e. the acceleration of the i^{th} joint $\ddot{\mathbf{q}}_i = \mathbf{a}^\top \mathbf{q} + \mathbf{b}^\top \dot{\mathbf{q}} + u$ for some coefficient vectors $\mathbf{a}$ and $\mathbf{b}$ and an uncertain $u \in [u_{\min}, u_{\max}]$. A model ignoring the nonlinear effects of dynamic parameters may suffer from greater over-approximation compared to a torque-based model, but will be quicker to calculate; fast methods exist for calculating reachable sets of linear systems [69]. We define the Minkowski sum.

Definition 4.3 Minkowski Sum

The Minkowski sum $\oplus$ is defined on two sets A and B so:

$$A \oplus B = \{a + b \,|\, a \in A, b \in B\}$$

When considering uncertain inputs, the system can be described by a set of linear differential inclusions of the form:

$$\begin{bmatrix} \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \end{bmatrix} \in \begin{bmatrix} 0 & I \\ C & D \end{bmatrix} \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix} \oplus \begin{bmatrix} [0,0] \\ [u_{\max}, u_{\min}] \end{bmatrix}, \tag{9}$$

where C and D are matrices of coefficients to be determined; I is an identity matrix of proper dimension.

An acceleration model which exploits the correlation between joint position or velocity and range of accelerations would be preferable, as this can combine the accuracy and insight offered by the torque model with the speed of calculation of the linear model. The dependence of joint torque on joint angle is well-known and studied for the single and multi-joint cases (e.g. [70]). The dependency in a single joint is due to muscle physiology and several models exist [71] whereas multi-joint interdependencies could be due to muscle-on-muscle impingement [72] and intermuscular force transmission, among other reasons [70]. From observation of our data, relationships between joint positions and accelerations can be observed, but less so between joint velocities and accelerations. The correlations over the entire data set can be seen in Table. 5 and an example of dependency from one movement can be seen in Fig. 25.

Table 5: Correlation between joint positions and velocities, and acceleration for 4-DOF Model

r^2	q_1	q_2	q_3	q_4	$\dot{q}_1$	$\dot{q}_2$	$\dot{q}_3$	$\dot{q}_4$
$\ddot{q}_1$	-0.12	-0.00	-0.02	0.03	-0.00	0.04	-0.00	0.06
$\ddot{q}_2$	-0.00	-0.27	0.08	-0.02	-0.09	0.00	-0.07	0.10
$\ddot{q}_3$	-0.02	0.02	-0.16	0.00	0.00	0.02	0.00	-0.03
$\ddot{q}_4$	0.06	-0.02	0.00	-0.20	-0.11	-0.07	0.06	-0.00

Furthermore, having modeled the arm very simply as two rods with even mass distribution, we find the Coriolis and centrifugal terms to be small in comparison to inertia terms. We therefore disregard dependency on the joint velocities, set the lower right sub-matrix D of the Jacobian matrix in (9) to a matrix of zeros. Note that this is still over-approximative, simply independent of $\dot{q}$, as the interval $[u_{\max}, u_{\min}]$ must still account for all accelerations observed in the data.

We populate the C sub-matrix. At our disposal we have the motion capture data from Sec. 4.3, from which we can extract, at all points in time, the acceleration of each joint $\ddot{q}_i(t)$ and the corresponding joint positions $\mathbf{q}(t)$. The accelerations must be within the interval in

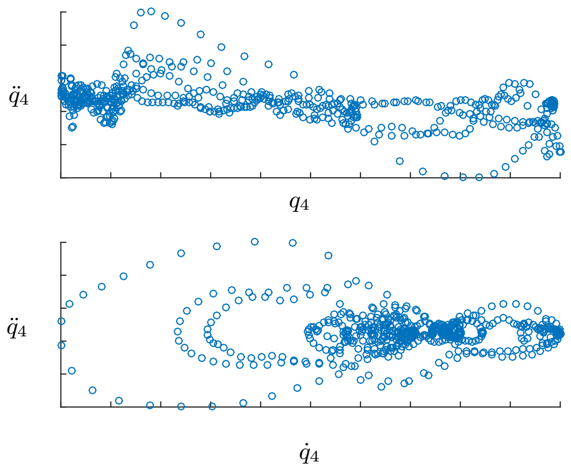


Figure 25: Dependence of $\ddot{q}_4$ (y-axis) on q_4 (x-axis top figure) and $\dot{q}_4$ (x-axis bottom figure) for a typical movement. The top figure exhibits negative linear correlation whereas there is no correlation in the bottom figure.

(9), i.e.

$$\ddot{q}_i(t) \in \sum_{j=1}^{j=n} C_{i,j} q_j(t) \oplus [u_{\min,i}, u_{\max,i}], \tag{10}$$

where $[u_{\min,i}, u_{\max,i}]$ is the i^{th} entry of $[u_{\min}, u_{\max}]$. We wish to determine the coefficient values of C such that the range of the interval vector $[u_{\max}, u_{\min}]$ is as narrow as possible, as a smaller range of accelerations leads to a slower-growing reachable set. One pathological solution is to set all coefficients to zero, meaning that $[u_{\max,i}, u_{\min,i}]$ is the interval of the maximum and minimum accelerations over the entire data set; this is not the smallest interval possible. To determine the optimal solution we first recall the definition of hyperplane and half-space:

Definition 4.4 HYPERPLANE *[73, p. 2] A hyperplane is a set which may be defined as* $H = \{x \in \mathbb{R}^k | n^\top x = \alpha\}$ *for suitable* $n \in \mathbb{R}^k$ *(the normal vector) and* $\alpha \in \mathbb{R}$.

Observe that this is a generalization of the plane to k-dimensional space.

Definition 4.5 HALF-SPACE *[73, p. 2] An open half-space is defined as* $H = \{x \in \mathbb{R}^k | n^\top x < \alpha\}$. *A closed half-space is defined as* $H = \{x \in \mathbb{R}^k | n^\top x \leq \alpha\}$.

Let C_i be the i^{th} row of C and let k be the number of joints, i.e. the dimension of $\mathbf{q}$. We examine the dependence of *all* joint positions on only one joint acceleration, and we do this for each joint $i \in \{1, ...k\}$ therefore we construct k data sets $\mathcal{S}_i, i \in \{1, ...k\}$, where the data points are values of $[\mathbf{q}^\top, \ddot{q}_i]^\top$ at every point in time over the entire motion capture data. If

$\mathbf{s}_i \in \mathbb{R}^{k+1}$ is an element of $\mathcal{S}_i$, it is a vector of $[\mathbf{q}^\top, \ddot{q}_i]^\top$ at some point in time. From these data sets, our goal is to find C such that $u_{\max} - u_{\min}$ is minimized.

We can rearrange Eq. (10) to define

$$\begin{aligned} u_{\max,i} &= max\{\begin{bmatrix} -C_i \,, 1 \end{bmatrix} \mathbf{s}_i | \mathbf{s}_i \in \mathcal{S}_i\}, \\ u_{\min,i} &= min\{\begin{bmatrix} -C_i \,, 1 \end{bmatrix} \mathbf{s}_i | \mathbf{s}_i \in \mathcal{S}_i\}. \end{aligned} \tag{11}$$

We observe that Eq. (11) is equivalent to the condition that all points lie in the intersection of the two closed half-spaces,

$$\begin{aligned} \mathcal{H}_u &= \{\, \mathbf{z} \,|\, [-C_i \,, 1\,](\mathbf{z}) \leq u_{\max,i}\}, \\ \mathcal{H}_l &= \{\, \mathbf{z} \,|\, [-C_i \,, 1\,](\mathbf{z}) \geq u_{\min,i}\}. \end{aligned}$$

Lemma 4.1 *Given a set of points $\mathcal{S}, \mathbf{s}_i \in \mathbb{R}^{k+1}$. Where $\mathbf{c} \in \mathbb{R}^k$ is a vector we define the function:*

$$\begin{aligned} m_1(\mathbf{c}) &= max\{[-\mathbf{c}^\top \,, 1\,]\mathbf{s}_i | \mathbf{s}_i \in \mathcal{S}\} \\ m_2(\mathbf{c}) &= min\{[-\mathbf{c}^\top \,, 1\,]\mathbf{s}_i | \mathbf{s}_i \in \mathcal{S}\} \\ m(\mathbf{c}) &= |m_1(\mathbf{c}) - m_2(\mathbf{c})| \end{aligned}$$

and the hyperplanes:

$$\begin{aligned} H_1(\mathbf{c}) &= \{\, \mathbf{z} \,|\, [-\mathbf{c}^\top \,, 1\,]\,\mathbf{z} = m_1(\mathbf{c})\} \\ H_2(\mathbf{c}) &= \{\, \mathbf{z} \,|\, [-\mathbf{c}^\top \,, 1\,]\,\mathbf{z} = m_2(\mathbf{c})\} \end{aligned}$$

If there are at least $k+1$ points in $\mathcal{S}$, a necessary condition for $m(\mathbf{c})$ to be a strict minimum over $\mathbf{c}$ is that there exist two nonempty sets $\mathcal{V}_1 = \{\mathbf{s} | \mathbf{s} \in \mathcal{S}, \mathbf{s} \in H_1(\mathbf{c})\}$ and $\mathcal{V}_2 = \{\mathbf{s} | \mathbf{s} \in \mathcal{S}, \mathbf{s} \in H_2(\mathbf{c})\}$ and $|\mathcal{V}_1| + |\mathcal{V}_2| \geq k+1$.

The proof is omitted for brevity. In order to find the global minimum, we implement Alg. 3: in lines 2 and 3, we construct the data sets $\mathcal{S}_i$ for each joint acceleration, and then the convex hull $CH(\mathcal{S}_i)$; since points inside the convex hull can never be boundary points of $\mathcal{S}_i$, they can never lie on the hyperplanes and we can disregard them from our search. In line 4 we loop through all sets of i points in $CH(\mathcal{S}_i)$ which define a bounding i-dimensional space of the convex hull, and in line 5 we find the complementary $k-i+1$-dimensional spaces on $CH(\mathcal{S}_i)$; together, these constitute $k+1$ points on the boundary of $\mathcal{S}_i$. For example in 3-D, we take parallel hyperplanes defined by pairs of line segments on $CH(\mathcal{S}_i)$, and then parallel hyperplanes defined each by one face and one point.

We check that these $k+1$ points are unique in line 6, and then find the hyperplanes defined by these points in line 7. In lines 8-11 we find $[u_{\min,i}, u_{\max,i}]$ and update the optimal bounding hyperplanes if the new value is a minimum. We can therefore populate the C matrix in Eq. (9) with the normals of the k pairs of hyperplanes.

Algorithm 3 Find Minimum Bounding Hyperplanes

Input: Sample points of $\mathbf{q}, \ddot{\mathbf{q}}$ in k dimensions

Output: $[u_{\min,i}, u_{\max,i}]$, normals to hyperplanes C

```
1: find convex hull of points
2: for $i < k$ do
3:   find $\mathcal{S}_i$ and $CH(\mathcal{S}_i)$
4:   for all $i$-spaces in $CH(\mathcal{S}_i)$ do
5:     for all $k+1-i$-spaces in $CH(\mathcal{S}_i)$ do
6:       if all $k+1$ points unique then
7:         Evaluate hyperplanes defined by these points
8:         Calculate $[u_{\min,i}, u_{\max,i}]$
9:         if $[u_{\min,i}, u_{\max,i}]$ is new minimum then
10:          Update optimal $\mathbf{c}$, $[u_{\min,i}, u_{\max,i}]$
11:        end if
12:      end if
13:    end for
14:  end for
15: end for
```

This exhaustive method quickly becomes impractical in higher-dimensional spaces, so for the 4-DOF model, only the relationship between acceleration and links 2, 3 and 4 are evaluated, as these exhibit the highest correlations, see Fig. 5. In the 3-DOF model the relationship between acceleration and all joints is evaluated. More scalable methods exist, e.g. [62] for finding sub-optimal solutions, however, as this computation is offline we have no time constraints, thus finding the global minimum is worth the extra computation time to reduce the size of $[u_{\min}, u_{\max}]$ as much as possible.

Methods for determining reachable sets of a linear first order system of differential inclusions for any time interval, given an initial set, are well known and fast [69]. We are now able to determine the reachable set of the kinematic chain given only the initial set of states, found online by inverse kinematics, and the dynamics, determined in advance.

4.6 Representation in Space

Representing a reachable set in the joint space presents challenges due to the highly nonlinear forward kinematics of the model as well as uncertainty in the length of the links. We address these challenges by splitting the 4-DOF model into two sub-chains of 2 rotational degrees of freedom each, *upper arm* and *forearm*, as described in Sec. 4.4.1. These chains appear as shown in Fig. 27 and are identical except for the length of the link. We find polyhedra that enclose the potential occupancy of upper arm and forearm in their own coordinate systems, and from these find another polyhedron that encloses the forearm in the coordinate system of the shoulder accounting for uncertainty in the position and orientation of the elbow as in Fig. 26. The 3-DOF model has only 2 rotational degrees of freedom and can be enclosed in one polyhedron. For clarity, ${}^{a}\mathbf{p}$ means the vector (or set of vectors) of the point (or set of points) $\mathbf{p}$ in coordinate system K_a.

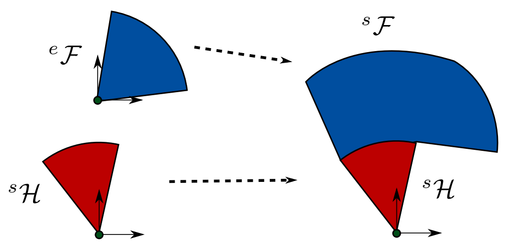


Figure 26: Occupancies of the upper arm and forearm; left: calculated apart in their own coordinate systems, right: joined together to the occupancy of the entire arm

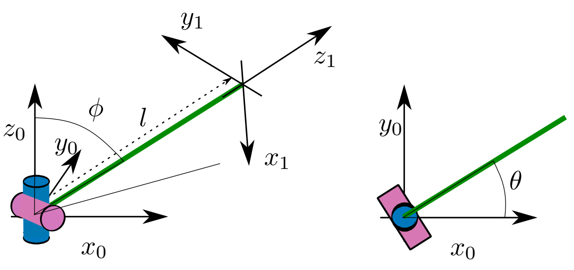


Figure 27: The kinematics of one subchain

The rest of this section describes and proves the conservativeness of the method for enclosing the arm in convex polyhedra. In Sec. 4.6.1 we detail assumptions made on the link shape and examine the forward kinematics of the sub-chain, showing that the possible endpoints of the link lie within an area on the surface of a sphere with the origin at the base of the sub-chain. In Sec. 4.6.2 and 4.6.3 we show in detail how the occupancy of a single subchain is enclosed, given a reachable set of joint angles. In Sec. 4.6.4 we show how this is extended to two subchains in the case of the 4-DOF model, and how to account for uncertainty in the length of the links. Finally, in Sec. 4.6.5 and 4.6.6 we present alternative methods of representation for comparison in Sec. 4.7

4.6.1 Forward Kinematics and Definition of Enclosing Polytope

We examine the forward kinematics of the subchain and define the problem of finding an *enclosing polytope*. The following definitions are useful in this and the next section.

A polyhedron that is closed (i.e. it contains its boundaries) and bounded (i.e., its volume is finite and contains no rays) can be called a *polytope*. A polytope P is the nonempty intersection of a finite set $S := \{S^1, \dots, S^q\}$ of halfspaces:

Definition 4.6 Halfspace Representation of Polytope *For q halfspaces, a convex polytope P is the set:*

$$P = \left\{ x \in \mathbb{R}^n \middle| Cx \leq d \right\},$$

where $C \in \mathbb{R}^{q \times n}, d \in \mathbb{R}^{q \times 1}$.

Each polytope can also be defined by the convex hull of a finite set of vertices $v^i \in \mathbb{R}^n$:

Definition 4.7 Vertex Representation of Polytope *For r vertices $v^i \in \mathbb{R}^n$, a convex polytope P is the set $P = CH(v^1, \dots, v^r)$, with*

$$CH(v^1, \dots, v^r) = \left\{ \sum_{i=1}^{r} \alpha^i v^i \middle| \alpha^i \in \mathbb{R}_{\geq 0}, \sum_{i=1}^{r} \alpha^i = 1 \right\}.$$

We use the following notation: a closed interval between $a_1, a_2 \in \mathbb{R}$ is denoted $[a_1, a_2]$; an open interval (a_1, a_2). The closed line segment between two points $\mathbf{p}_1, \mathbf{p}_2 \in \mathbb{R}^k$ is $\langle \mathbf{p}_1, \mathbf{p}_2 \rangle$.

Define the function g to be the forward kinematic transformation applied to a point ${}^1\mathbf{p}$ in the end-effector coordinate system K_1 resulting in ${}^0\mathbf{p}$ in the base coordinate system K_0:

$$g(\theta, \phi, l, {}^1\mathbf{p}) = {}^0\mathbf{p},$$

where θ is the first joint angle, ϕ is the second joint angle and l is the length of the link. This transformation is illustrated in Fig. 28.

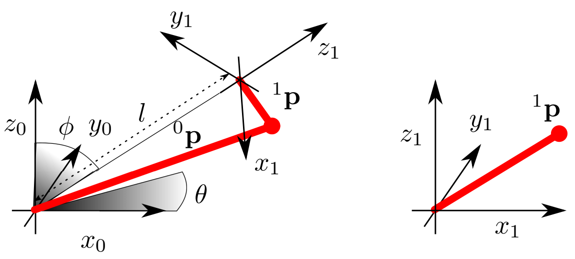


Figure 28: The transformation g. Where ${}^1\mathbf{p}$ is a vector in the end-effector coordinate system K_1, (right) the transformation $g(\theta, \phi, l, {}^1\mathbf{p})$ is the vector now in the base coordinate system K_0 (left).

Definition 4.8 Enclosing Polytope

Given a kinematic subchain as in Fig. 27 with uncertain joint angles $\theta \in [\theta_{\min}, \theta_{\max}]$ and $\phi \in [\phi_{\min}, \phi_{\max}]$, and cylindrical link of uncertain length $l_{\min} \leq l \leq l_{\max}$ and radius r, an enclosing polytope is a polytope $\mathcal{P}$ with respect to K_0 which encloses the possible occupancy of the link, i.e.

$$\mathcal{P} \supseteq \left\{ g(\theta, \phi, l, \mathbf{p}) \;\middle|\; \begin{aligned} \theta \in\ & [\theta_{\min}, \theta_{\max}], \\ \phi \in\ & [\phi_{\min}, \phi_{\max}], \\ l \in\ & [0, l_{\max}], \\ \mathbf{p} \in\ & r[\cos(\alpha), \sin(\alpha), 0]^\top, \\ \alpha \in\ & [-\pi, \pi], \\ r \in\ & [0, r_{\max}] \end{aligned} \right\} \tag{12}$$

The volume of the link is a cylinder of length l and radius r. This can be considered to be the Minkowski sum (see Def. 4.3) of a line segment of length l and a circle of radius r on a plane normal to that line. We will first consider the link shape to be a line segment and then extend the polytope obtained at the end of the algorithm accordingly to account for the radius of the cylinder.

This leads to the following simplification: if the start and end-points of a line segment are enclosed in a convex shape, the entire segment between them is enclosed, by the definition of convexity. The start point of the link is at the origin, thus we need only to enclose the endpoint of the link, $g(\theta, \phi, l_{\max}, \mathbf{p})$, and the origin.

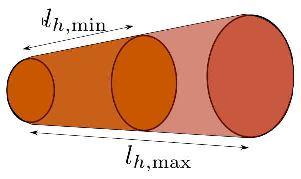


Figure 29: Smaller upper arm lengths $l_{h,\min}$ are contained within the reachable occupancy calculated from the maximum upper arm length $l_{h,\max}$.

We additionally account for parametric uncertainty in the length, as a link of length $l_{\min} < l_{\max}$ will be enclosed by the occupancy of a link of length $l_{\max}$, see Fig. 29. In the following sections, we will assume the arguments l and ${}^1\mathbf{p}$ to be fixed values, and define

$$g_{occ} = \{g(\theta, \phi, l, {}^1\mathbf{p}) | \theta \in [\theta_{\min}, \theta_{\max}], \phi \in [\phi_{\min}, \phi_{\max}]\}$$

Eq. 12 then becomes:

$$\mathcal{P} \supseteq CH(g_{occ}\,,\ \mathbf{0})\ \oplus\ Y(r),$$

where $Y(r)$ is the over-approximation to be added due to the radius of the cylinder. From Fig. 28 we observe that $|{}^{0}\mathbf{p}| = |[0,0,l]^{\top} + {}^{1}\mathbf{p}|$, which is fixed, therefore all values of g_{occ} lie on the surface of a sphere $\mathcal{S}$ of radius $|[0,0,l]^{\top} + {}^{1}\mathbf{p}|$. In the following section we show how to construct a polytope enclosing this area on the surface of the sphere as well as the origin.

4.6.2 Enclosing a single link

In this section we show how g_{occ} can be projected to a plane and bounded by a closed curve k' in the plane, then this curve projected back to a curve k on the sphere $\mathcal{S}$ to bound g_{occ} on the surface of $\mathcal{S}$. We then show how k can be extended to a polytope bounding both g_{occ} and the origin of $\mathcal{S}$. We define the *circle of a sphere* and the *spherical polygon*:

Definition 4.9 Circle of a Sphere *[74]:*

*The section of the surface of a sphere g made by any plane is called a **circle of a sphere**. If the plane passes through the centre of the sphere, this circle G is called a **great circle (of a sphere)**, see Fig. 30; an arc of a great circle is called a **geodesic**. Analogously to lines, the geodesic between two points on a sphere $\mathbf{s}_1$ and $\mathbf{s}_2$ is denoted $\langle \mathbf{s}_1, \mathbf{s}_2 \rangle$*

*The **axis** z of any circle of a sphere is that diameter of the sphere which is perpendicular to the plane of the circle. The extremities of the axis are called the **poles** y_1 and y_2.*

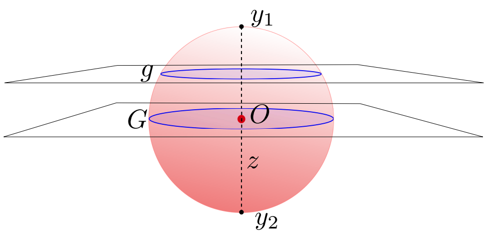


Figure 30: A circle of a sphere g and a great circle G with axis z and poles y_1 and y_2.

Definition 4.10 Spherical Polygon *[74]:*

*A spherical polygon is a closed curve on the surface of a sphere formed by two or more arcs of great circles (geodesics). We call a spherical polygon **convex** if, for any two points inside the polygon, the polygon contains (at least) one geodesic between these points [75].*

*The union of rays drawn from the origin O of the sphere through the points of the spherical polygon k constitutes the corresponding **polyhedral angle** K [75], see Fig. 31. If the spherical polygon is convex, the polyhedral angle is equivalently the intersection of the halfspaces defined by the arcs of the spherical polygon.*

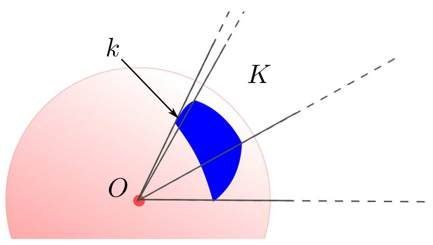


Figure 31: Spherical polygon k and corresponding polyhedral angle K.

Because it is easier to work in two dimensions than three, we use the gnomonic projection, which defines a bijection between the surface of a closed halfsphere to the *projective plane* as explained in [76, p. 145]. The projective plane $\mathbb{PR}^2$ is the Euclidean plane $\mathbb{R}^2$ extended by a line at infinity representing the projection of the edge of the halfsphere.

Definition 4.11 Gnomonic Projection *[77]*

The gnomonic projection is a projection of points on the surface of a sphere $\mathcal{S}$ from the origin O of the sphere onto the projective plane $\mathcal{P}$ at a tangent to the sphere at y. Where (ρ, ϑ, ψ) are the spherical coordinates (radius, azimuth and polar angle) of a point p on $\mathcal{S}$, the polar coordinates (r, φ) of p' in the projective plane are:

$$
\begin{aligned}
r &= \frac{\rho}{\tan(\psi)} \\
\varphi &= \vartheta
\end{aligned}
\tag{13}
$$

This has the following properties:

1. *any great circles G is projected to a straight line G',*
2. *circles a of $\mathcal{S}$ with a pole of y are projected to circles a' centered on y',*
3. *circles b of $\mathcal{S}$ with a pole on the great circle parallel to $\mathcal{P}$ are projected to hyperbolae b',*
4. *only one hemisphere can be projected onto the plane.*

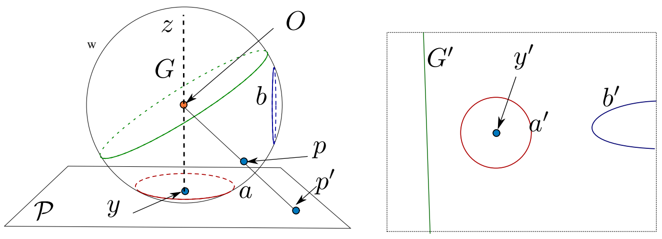


Figure 32: Circles of $\mathcal{S}$ in the gnomonic projection.

As great circles are defined by lines in the gnomonic projection, for any two points a and b and their corresponding projections a' and b', the geodesic $\langle a, b\rangle$ corresponds to the line segment $\langle a', b' \rangle$ and vice-versa, so a convex polygon on the plane is a convex spherical polygon on $\mathcal{S}$.

Our method will be such: after projecting g_{occ} to a plane, we find a convex polygon bounding it, project this back to a convex spherical polygon on the sphere and find the corresponding polyhedral angle, which by definition bounds g_{occ} and the origin. We then intersect this with hyperplanes tangent to the sphere to form a bounded polyhedron.

We now find g'_{occ}, the projection of g_{occ} onto a suitable plane. Given a point ${}^0\mathbf{a} = g(\theta, \phi, l, {}^1\mathbf{p})$ on the sphere $\mathcal{S}$ of radius $|{}^0\mathbf{p}|$, the point ${}^0\mathbf{b} = g(\theta + \Delta\theta, \phi, l, {}^1\mathbf{p})$ is ${}^0\mathbf{a}$ rotated around the axis of joint 1 by $\Delta\theta$ and the point ${}^0\mathbf{c} = g(\theta, \phi + \Delta\phi, l, {}^1\mathbf{p})$ is ${}^0\mathbf{a}$ rotated around the axis of joint 2 by $\Delta\phi$. Therefore the locus of points:

$$U(\theta) = \{g(\theta, \phi, l, \mathbf{p}) | \phi \in [\phi_{\min}, \phi_{\max}]\}$$

is the arc of a circle of $\mathcal{S}$ with the axis of joint 2, which is perpendicular to the axis of joint 1, and

$$\begin{aligned} g_{occ} =& \{g(\theta, \phi, l, {}^1\mathbf{p}) | \theta \in [\theta_{\min}, \theta_{\max}], \phi \in [\phi_{\min}, \phi_{\max}]\} \\ =& \{U(\theta) | \theta \in [\theta_{\min}, \theta_{\max}]\} \end{aligned}$$

is a union of such arcs of circles of $\mathcal{S}$ rotated around the axis of joint 1 by an angle $\theta \in [\theta_{\max}, \theta_{\min}]$

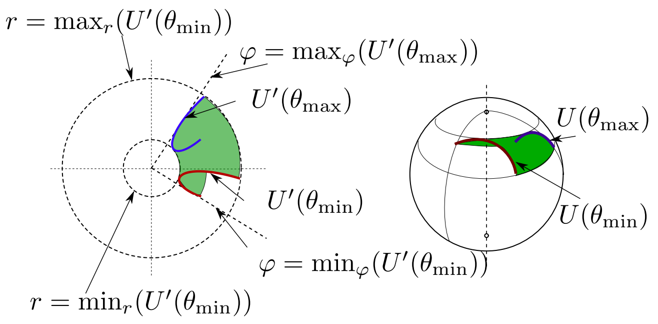
We take the gnomonic projection of $U(\theta)$ onto a plane perpendicular to the first joint axis, and obtain $U'(\theta)$, which is an arc of a hyperbola by condition 3 of Def. 4.11; this is illustrated in Fig. 33. We work in polar coordinates in the plane, i.e. every point $\mathbf{u} \in U'(\theta)$ can be expressed as a radius and angle, $\mathbf{u} = (r, \varphi)$. We define the functions $\max_\varphi(U'(\theta)), \min_\varphi(U'(\theta))$, and $\min_r(U'(\theta)), \max_r(U'(\theta))$, which return the minimum and maximum values of φ and r in $U'(\theta)$, respectively.

By Eq. 13, $U'(\theta + \Delta\theta) = \{\mathbf{u} + (0, \Delta\theta) | \mathbf{u} \in U'(\theta)\}$, i.e. a rotation $\Delta\theta$ on the sphere leads to a rotation $\Delta\varphi$ in the projection. Therefore

$$\begin{aligned} \max\nolimits_\varphi(\{U'(\theta) | \theta \in [\theta_{\min}, \theta_{\max}]\}) = \max\nolimits_\varphi(U'(\theta_{\max})) \\ \min\nolimits_\varphi(\{U'(\theta) | \theta \in [\theta_{\min}, \theta_{\max}]\}) = \min\nolimits_\varphi(U'(\theta_{\min})), \end{aligned}$$

and g'_{occ} is bound by the level sets $\varphi = \max_\varphi(U'(\theta_{\max}))$ and $\varphi = \min_\varphi(U'(\theta_{\min}))$.

Furthermore, since $U'(\theta)$ is independent of r, g'_{occ} is also bound (without loss of generality) by the level sets $r = \min_r(U'(\theta_{\min}))$ and $r = \max_r(U'(\theta_{\min}))$.



$r = \max_r(U'(\theta_{\min}))$
$\varphi = \max_\varphi(U'(\theta_{\max}))$
$U'(\theta_{\max})$
$U'(\theta_{\min})$
$\varphi = \min_\varphi(U'(\theta_{\min}))$
$r = \min_r(U'(\theta_{\min}))$
$U(\theta_{\max})$
$U(\theta_{\min})$

Figure 33: Illustration of g_{occ} (right) and projection g'_{occ} (left)

Having determined the shape of g'_{occ}, we proceed to bound it with a convex polygon.

4.6.3 Obtaining the Enclosing Polytope

The construction of the enclosing polytope is detailed in Alg. 4. In constructing the polygon in $\mathcal{P}$ we must consider three cases. We call the *equator* that great circle which is perpendicular to the axis of joint 1, and we take hemisphere to mean a closed hemisphere, i.e. one half of the sphere bisected by the equator, including the equator.

Algorithm 4 `enclosingPolytope`$(\Theta, \Phi, l, \mathbf{p}, r)$

Input: interval Φ, interval Θ, link length $l_{\max}$, vector $\mathbf{p}$, cylinder radius $r_{\max}$

Output: Polytope(s) enclosing $\{g(\phi, \theta, l, \mathbf{p}) | \phi \in \phi, \theta \in \theta\}$

```
1:  if case 1 then
2:      Find $\theta_{\max}, \theta_{\min}$ in projection
3:      Find $r_{\max}, r_{\min}$ in projection, intersection points
4:  else
5:      Find $\theta_{asy}$ {WLOG, $\varphi+$}
6:      Find $\theta_{\min,1}$ and $\theta_{\min,2}$ in projection
7:      Find $r_{\max,1}$ and $r_{\max,2}$ in projection and intersection points
8:  end if
9:  if $\theta_{\max} - \theta_{\min} > \pi$ then
10:     Find $T_1$ and $T_2$ and convex hull
11: end if
12: Project intersection points back to sphere and construct spherical angle {These are normal vectors}
13: Bound with hyperplanes tangent to surface of $\mathcal{S}$
14: Enlarge to account for radius of cylinder
```

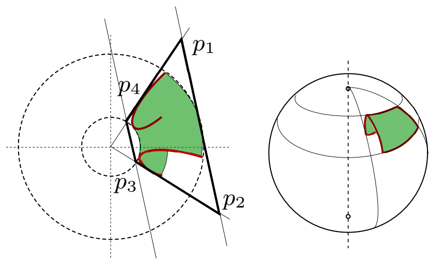


Figure 34: Case 1: g_{occ} lies on one hemisphere

Case 1: g_{occ} lies on one hemisphere In the case that g_{occ} lies entirely in one hemisphere, g'_{occ} lies entirely on the projective plane tangent at the pole of that hemisphere. The lines $\varphi = \max_\varphi(U'(\theta_{\max}))$, $\varphi = \min_\varphi(U'(\theta_{\min}))$, $r = \min_r(U'(\theta_{\min}))$ and $r = \max_r(U'(\theta_{\min}))$ form an annular segment, which can be enclosed in a convex quadrilateral (see Fig. 34) the vertices of which can be calculated and projected back to the sphere. This is performed in lines 2-3 of Alg. 4.

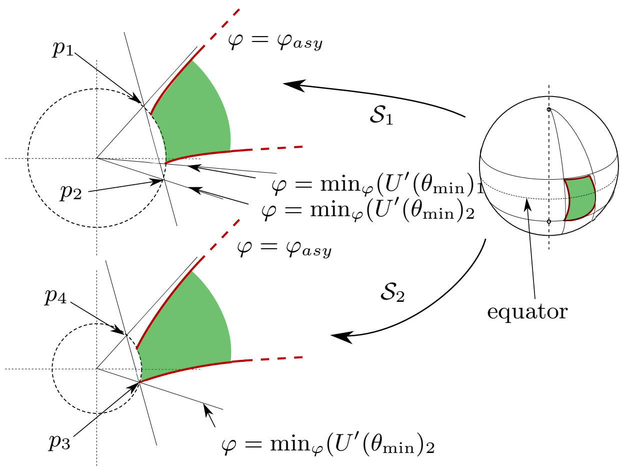

p_1
$\varphi = \varphi_{asy}$
$\mathcal{S}_1$
p_2
$\varphi = \min_\varphi(U'(\theta_{\min})_1$
$\varphi = \min_\varphi(U'(\theta_{\min})_2$
$\varphi = \varphi_{asy}$
p_4
$\mathcal{S}_2$
equator
p_3
$\varphi = \min_\varphi(U'(\theta_{\min})_2$

Figure 35: Case 2: g_{occ} lies on both hemispheres

Case 2: g_{occ} lies on both hemispheres In the case that g_{occ} is bisected by the equator (lines 4-7 of Alg. 4), we split the arc $U(\theta)$ over the two hemispheres $\mathcal{S}_1$ and $\mathcal{S}_2$ into into $U_1(\theta) \subseteq \mathcal{S}_1$ and $U_2(\theta) \subseteq \mathcal{S}_2$, so that $g_{occ,1} = \{U(\theta)_1 | \theta \in [\theta_{\min}, \theta_{\max}]\}$ and $g_{occ,2} = \{U(\theta)_2 | \theta \in [\theta_{\min}, \theta_{\max}]\}$, and project onto planes at the y_1 and y_2 poles as shown in Fig 35. In the respective planes, we observe that the hyperbolae $U'(\theta)_i$ extend to intersect the line at infinity (recall that we work in the projective plane) which corresponds to the equator on $\mathcal{S}$. Without loss of generality, the hyperbola extends to infinity on the side of positive φ as in Fig 35. Let $\varphi = \varphi_{asy}$ be the asymptote to the hyperbola $U'(\theta_{\max})$; this bounds g_{occ} on the positive φ side.

φ_{asy} can be obtained from the intersection of $U(\theta_{\max})$ and the equator. To bound φ on the negative side, we take:

$$\varphi = \min(\min_{\varphi}(U'(\theta_{\min})_2), \min_{\varphi}(U'(\theta_{\min})_1)). \tag{14}$$

Finally, $r = \min_r(U'_1(\theta_{\min}))$ and $r = \min_r(U'_2(\theta_{\min}))$ are calculated; the intersection points with $\varphi = \varphi_{asy}$, p'_1 and p'_2, and with (14), p'_4 and p'_3, are calculated. Lines l_1 and l_2 passing through these points bound g_{occ} on the side away from each pole y_1 and y_2. The intersections of these lines are projected back to $\mathcal{S}$ to define the corners of the spherical polygon k.

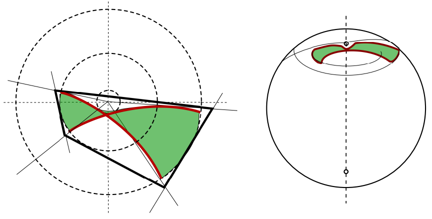


Figure 36: Case 3: convex hull of g_{occ} intersects the pole

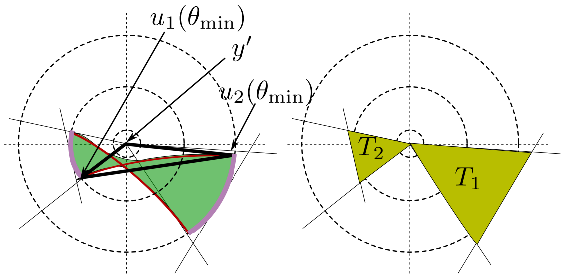


Figure 37: Constructing a convex polygon in case 3

Case 3: $\max\varphi(U'(\theta_{\max})) - \min_{\varphi}(U'(\theta_{\min})) > \pi$ Finally, in either of these cases, the angle $\max\varphi(U'(\theta_{\max})) - \min_{\varphi}(U'(\theta_{\min}))$ may be greater than π, lines 9-11 of Alg. 4. In this case the approach in case 1 and 2 no longer yields a convex polygon. We therefore bound each $U'(\theta)$ in a triangle as shown in Fig. 37, with vertices $u_1(\theta)$, $u_2(\theta)$ and y'. $u_1(\theta)$ and $u_2(\theta)$ lie on arcs around y' subtending $\theta_{\max} - \theta_{\min}$ which may be bounded by triangles T_1 and T_2.

Lemma 4.2 *let $Z = \{z_1, z_2, ...z_n\}$ be a set of points and $\mathcal{P} = \{\wp_1, \wp_2, ...\wp_n\}$ be a set of polytopes. Then*

$$\forall i \in \{1, ...n\}, z_i \in \wp_i \implies CH(Z) \subseteq CH(\mathcal{P})$$

Proof 3 *If $\forall i, z_i \in \wp_i \subseteq CH(\mathcal{P})$, then $Z \subseteq CH(\mathcal{P})$. From [73, p. 14], $CH(Z)$ is the intersection of all convex sets containing Z, and $CH(\mathcal{P})$ is such a convex set. Therefore $CH(Z) \subseteq CH(\mathcal{P})$.*

Since $U(\theta) \subseteq CH(u_1(\theta), u_2(\theta), y')$ and $u_1(\theta) \subseteq T_1, u_2(\theta) \subseteq T_2$, by Lem. 4.2 $CH(u_1(\theta), u_2(\theta), y') \subseteq CH(T_1, T_2, y')$. The convex hull of two triangles and a point is straightforward to calculate; this bounds g_{occ}.

Having obtained a spherical polygon k and therefore a polyhedral angle K (line 12 of Alg. 4), we must bound K (which is unbounded) to obtain a polytope. In line 13 of Alg. 4 we intersect K with the halfspaces tangent to $\mathcal{S}$; we prove that this is now bounded:

Lemma 4.3 *Let k be a convex spherical polygon lying in one open hemisphere of sphere $\mathcal{S}$ and K be the polyhedral angle corresponding to it. At each vertex v_i of the spherical polygon, let the normal plane V_i define a halfspace $\mathcal{V}_i$. Then the polyhedron which is the intersection of K and halfspaces $\mathcal{V}_i$ is bounded.*

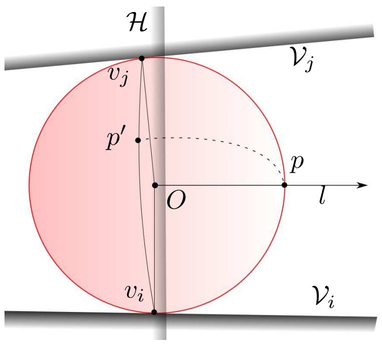


Figure 38: An illustration for proof of lemma 4.3. l is a ray, p is a point. By showing that the hyperplanes tangent to the sphere must have normals in a closed hemisphere opposite to l, the assumption that the convex polyhedron k lies in an open hemisphere is contradicted.

Proof 4 *Suppose not, then there exists a ray l contained within the polyhedron with initial point at any point inside the polyhedron, [75]. Without loss of generality, let this ray's initial point be the origin. Let the hyperplane normal to l be H; let the closed halfspace defined by this hyperplane and containing only the initial point of the ray be $\mathcal{H}$. In order that this ray does not intersect any hyperplane V_i, the normals to all V_i must be in $\mathcal{H}$, see Fig. 38. Recall that the normals to these halfspaces are the vectors of the vertices of the spherical polygon k, therefore all vertices of k lie in the intersection $\mathcal{H} \cap S$. Note that this is a closed hemisphere and therefore also a convex spherical polygon [75], therefore, for any two points on this hemisphere, e.g. vertices v_i and v_j, the geodesic between them $\langle v_i, v_j \rangle$ also lies in the hemisphere. Therefore both vertices and edges of k lie in this hemisphere.*

Let the intersection of the ray and S be p. Then because of the convexity of k, for any point p' on the boundary of k, the geodesic $[p, p']$ lies inside k. We have the condition that k lies in one open hemisphere. However, the entire boundary of $\mathcal{H}$, which is a great circle, lies in k, and this is a contradiction. Therefore the polyhedron cannot be unbounded.

Thus a polytope is formed which encloses the endpoints of the link and the origin; line 13 of Alg. 4. Finally, as the link is a cylinder, not a line segment, the radius of the link r is added to the polytope; see line 14 in Alg. 4. This is done in an over-approximative way by shifting the half-spaces defining the polyhedral angle outwards by the radius of the link and enlarging the current radius of the sphere $|[l,0,0]^\top + {}^1\mathbf{p}|$ to $\sqrt{|[l,0,0]^\top + {}^1\mathbf{p}|^2 + r^2}$. Another half-space H in Fig. 39 at a distance of r, entirely in the hemisphere not containing g_{occ}, is intersected with the resulting polytope, as otherwise the intersection point V extends too far away from the origin.

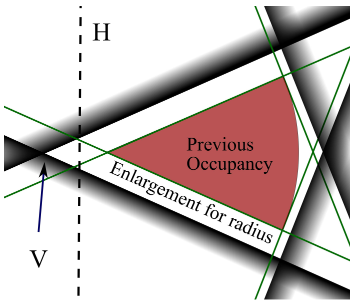


Figure 39: The general approach shown in 2-D: The halfspaces bounding the occupancy are extended to account for the radius of the link.

4.6.4 Application to Kinematic Models

3-DOF Model The 3-DOF model requires only one bounding polytope. The length of the link is taken as the supremum value of the 3rd, prismatic joint. The cylinder radius is taken to be the maximum radius, as justified in Sec. 4.4.2.

4-DOF Model The algorithm for the 4-DOF model is as follows. We first calculate the occupancy of the upper arm in K_s and of the forearm in K_e as in the previous section, which are ${}^s\mathcal{H}$ and ${}^e\mathcal{F}$ respectively. These are lines 1 and 2 in Alg. 5. We then construct two over-approximative polytopes ${}^s\mathcal{F}_{h,\min}$ and ${}^s\mathcal{F}_{h,\max}$ in which the occupancy of the forearm in K_s for an arm with minimum length upper arm and an arm with maximum length upper arm is contained. This is line 7 of Alg. 5. Finally, we take the convex hull of these two polytopes to obtain the polytope ${}^s\mathcal{F}$ which encloses the forearm for any upper arm length, in line 9 of Alg. 5. Together with ${}^s\mathcal{H}$, this encloses the human arm.

Constructing ${}^s\mathcal{F}_{h,\min}$ and ${}^s\mathcal{F}_{h,\max}$ We now present the method for obtaining ${}^s\mathcal{F}_{h,\min}$ and ${}^s\mathcal{F}_{h,\max}$ in lines 4-7 of Alg. 5. Having bounded the forearm movement in its own coordinate system K_e, we consider the forearm occupancy a static shape and no longer dependent on q_3

Algorithm 5 Finding bounding volumes of a two-link robot

Input: Reachable intervals of each joint: Q_1, Q_2, Q_3, Q_4; minimum and maximum segment lengths: $l_{h,\min}$, $l_{h,\max}$, $l_{f,\max}$; radius of link cylinders: r_h, r_f.

Output: Two convex polytopes ${}^s\mathcal{H}, {}^s\mathcal{F}$

```
1: ${}^s\mathcal{H}$=enclosingPolytope($Q_1, Q_2, l_{h,\max}, \mathbf{0}, r_h$) {upper arm in $K_s$}
2: ${}^e\mathcal{F}$=enclosingPolytope($Q_3, Q_4, l_{f,\max}, \mathbf{0}, r_f$) {Forearm in $K_e$}
3: for $l \in \{l_{h,\min}, l_{h,\max}\}$ do
4:     for each vertex ${}^e\mathbf{v}^{(i)}$ of ${}^e\mathcal{V}$ do
5:         ${}^s\mathcal{F}_h^{(i)}$ = enclosingPolytope($Q_1, Q_2, l, {}^e\mathbf{v}^{(i)}, 0$)$\backslash\mathbf{0}$
6:     end for
7:     ${}^s\mathcal{F}_h$ = convexHull(${}^s\mathcal{F}_h^{(i)}$)
8: end for
9: ${}^s\mathcal{F}$ = convexHull(${}^s\mathcal{F}_{h,\min}, {}^s\mathcal{F}_{h,\min}$)
```

and q_4. As the "static shape" is a polytope in K_e, it can be defined by its vertices ${}^e\mathbf{v}_i \in {}^e\mathcal{V}$ as in Def. 4.7, which are points in K_e. The occupancy in K_s of a point ${}^e\mathbf{v}_i$ in K_e, we recall, is $\{g(\theta, \phi, l, \mathbf{v}_i)\}$, which is over-approximated by following Alg. 4. We prove that the convex hull of all $\{g(\theta, \phi, l, {}^e\mathbf{v}_i)\}$ for ${}^e\mathbf{v}_i \in {}^es\mathcal{V}$ over-approximates the forearm occupancy in K_s. Let ${}^e\mathcal{V}$ be the vertices ${}^e\mathbf{v}^{(i)}$ of ${}^e\mathcal{F}$ and let CH be the convex hull operator. From Def. 4.7, the convex polytope is the convex hull of its vertices, therefore ${}^eCH(\mathcal{V}) = {}^e\mathcal{F}$.

Note that $g(\theta, \phi, l, \mathbf{p})$ is an affine transformation of $\mathbf{p}$ for constant θ, ϕ and l and therefore preserves convexity [73, p. 9]. Then:

$$\begin{aligned} &{}^e\mathbf{f} \in \mathrm{CH}({}^e\mathcal{V}) \\ &\implies g(q_1, q_2, l_h, {}^e\mathbf{f}) \in \mathrm{CH}(\{g(q_1, q_2, l_h, {}^e\mathbf{v}) | {}^e\mathbf{v} \in {}^e\mathcal{V}\}), \end{aligned} \tag{15}$$

for any q_1 and q_2. We calculate the set of polyhedra ${}^s\mathcal{W}$:

$$\begin{aligned} {}^s\mathcal{W} &= \{{}^s\mathbf{W}({}^e\mathbf{v}_i) | {}^e\mathbf{v}_i \in {}^e\mathcal{V}\} \\ {}^s\mathbf{W}({}^e\mathbf{v}) &\supseteq \left\{ g(\theta, \phi, l, {}^e\mathbf{v}) \;\middle|\; \begin{aligned} &\theta \in [q_{1,\min}, q_{1,\max}], \\ &\phi \in [q_{2,\min}, q_{2,\max}] \end{aligned} \right\} \end{aligned}$$

By Lemma 4.2:

$$\begin{aligned} &\forall i \in \{1, ...n\}, \{g(q_1, q_2, l_h, {}^e\mathbf{v}_i) \in {}^s\mathbf{W}({}^e\mathbf{v}_i) \implies \\ &\mathrm{CH}(\{g(q_1, q_2, l_h, {}^e\mathbf{v}) | {}^e\mathbf{v} \in {}^e\mathcal{V}\}) \subseteq CH({}^s\mathcal{W}) \end{aligned} \tag{16}$$

From Eq. (15) and 16,

$$\left\{ g(q_1, q_2, l_h, {}^e\mathbf{f}) \;\middle|\; \begin{array}{l} {}^e\mathbf{f} \in {}^e\mathcal{F}, \\ \theta \in [q_{1,\min}, q_{1,\max}], \\ \phi \in [q_{2,\min}, q_{2,\max}] \end{array} \right\} \in \mathrm{CH}({}^s\mathcal{W})$$

So, we choose ${}^s\mathcal{F} = {}^s\mathrm{CH}(\mathcal{W})$. In contrast to the single-link case, because we do not enclose the origin, we simply exclude the origin from the vertex representation of each ${}^s\mathbf{W} \in {}^s\mathcal{W}$. The proof that this polytope still encloses the boundary on the surface of the sphere is trivial and follows from the fact that the vertices of the spherical polygon k are on the sphere and that the sphere is a convex object.

Parametric uncertainty in first link In the single-link case, the uncertainty in link length is accounted for by the fact that the occupancy of smaller forearms is a subset of that of the maximal-length forearm (Fig. 29). However, the parametric uncertainty of the upper arm needs to be taken into account in the occupancy of the forearm. This subsection concerns itself with lines 3 and 9 in Alg. 5, in which the convex hull of the forearm with the minimum length upper arm ${}^s\mathcal{F}_{h,\min}$ and the forearm with the maximum length upper arm ${}^s\mathcal{F}_{h,\max}$ is taken. We prove that the polytope obtained encloses the forearm occupancy for all upper arm lengths. We drop the indicator of the coordinate system as we now work only in K_s.

Lemma 4.4 *Let $\mathcal{F}(l_h)$ be the occupancy of the forearm in the base coordinate system with a upper arm length l_h. Then the occupancy of the forearm for any $l_{h,\min} \leq l_h \leq l_{h,\max}$ lies in $\mathcal{F} = CH(\mathcal{F}(l_{h,\min}), \mathcal{F}(l_{h,\max}))$.*

Proof 5 *See Fig. 40. let $\mathbf{a}(l_{h,\max})$ and $\mathbf{a}(l_{h,\min})$ be the vectors from the shoulder to the elbow for given q_1 and q_2 and the minimum $l_{h,\min}$ and maximum $l_{h,\max}$ upper arm lengths respectively and let $\mathbf{b}$ be the vector from the elbow to an arbitrary point $\mathbf{p}(l_h)$ on the forearm, for given q_1, q_2, q_3 and q_4, such that $\mathbf{p}(l_h) = \mathbf{b} + a(l_h)$. We observe that $\mathbf{a}(l_{h,\max})$ and $\mathbf{a}(l_{h,\min})$ are collinear, so $\mathbf{a}(l_{h,\min}) = \lambda_{\min}\mathbf{a}(l_{h,\max}), \lambda_{\min} = \frac{l_{h,\min}}{l_{h,\max}}$. Thus, the position of the elbow for upper arm length $l_{h,\min} \leq l_h \leq l_{h,\max}$ lies on the line segment $\lambda\mathbf{a}(l_{h,\max}), \lambda_{\min} \leq \lambda \leq 1$.*

As the orientation of the forearm with respect to the shoulder (origin) is independent of upper arm length, $\mathbf{b}$ is independent of upper arm length and $\mathbf{p}(l_h)$ therefore lies on the line segment $\langle \mathbf{p}(l_{h,\max}), \mathbf{p}(l_{h,\min}) \rangle = \{\mathbf{b} + \lambda\mathbf{a}(l_{h,\max}) | \lambda_{\min} \leq \lambda \leq 1\}$. Then, as the convex hull of two points is the line segment between them, by Lemma 4.2, $\mathbf{p}(l_{h,\max}) \in \mathcal{F}(l_{h,\min}), \mathbf{p}(l_{h,\min}) \in \mathcal{F}(l_{h,\max}) \implies \langle \mathbf{p}(l_{h,\max}), \mathbf{p}(l_{h,\min}) \rangle \in CH(\mathcal{F}(l_{h,\min}), \mathcal{F}(l_{h,\max}))$.

Therefore any arbitrary point on the forearm $\mathbf{p}(l_h)$ *where* $l_{h,\min} \leq l_h \leq l_{h,\max}$*,* $\mathbf{p} \in CH(\mathcal{F}(l_{h,\min}), \mathcal{F}(l_{h,\max}))$

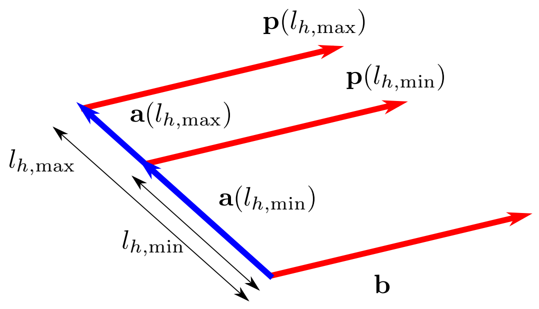


Figure 40: Diagram for proof of Lemma 4.4. Since $\mathbf{a}(l_{h,\min})$ is a scalar multiple of $\mathbf{a}(l_{h,\max})$, and $\mathbf{p}$ = $\mathbf{a}$ plus a fixed vector $\mathbf{b}$, enclosing $\mathbf{p}(l_{h,\min})$ and $\mathbf{p}(l_{h,\max})$ encloses all values of $\mathbf{p}$ in between.

We next present two existing methods from the literature for enclosing links of a kinematic chain, for comparison to the method described above.

4.6.5 Sampling Approach

The idea of *Reachable Volumes* for enclosing the occupancy of kinematic chains is presented in [78]; the computation of these volumes uses a sampling-based approach, but does not formally guarantee that all reachable joint positions lie within the reachable volume. Such sampling-based methods are computationally expensive, especially if the sample density is high, and therefore best suited for offline applications.

In our implementation, each link is represented by a collection of occupancy samples in its own coordinate system. In the 4-DOF model, the upper arm is represented by the set of occupancy samples $\mathcal{V}_H = \{h_1, ..., h_p\}$ and the forearm by the set $\mathcal{V}_F = \{f_1, ...f_q\}$, where $h_i, f_i \in \mathbb{R}^3$. We sample the reachable set in the joint space, and find the sets of forward kinematics transformation matrices $\mathcal{T}_H = \{{}^e\mathbf{T}_s^1, ..., {}^e\mathbf{T}_s^r\}$ and $\mathcal{T}_F = \{{}^w\mathbf{T}_s^1, ..., {}^w\mathbf{T}_s^s\}$, where ${}^e\mathbf{T}_s^j, {}^w\mathbf{T}_s^j \in \mathbb{R}^{4\times4}$ are the transformation matrices from the base coordinate system K_s to the elbow coordinate system K_e and the end-effector coordinate system K_w respectively, for each joint space sample $j \in \mathcal{R}_q$. We apply these transformation matrices to the occupancy samples, obtaining:

$$\mathcal{H} = \{{}^2\mathbf{T}_0^j h_i | h_i \in H, {}^e\mathbf{T}_s^j \in T_h\}$$
$$\mathcal{F} = \bigcup_{i=1}^{8} \bigcup_{j=1}^{p} {}^w\mathbf{T}_s^j f_i$$

Finally, for the 4-DOF model, we find the two convex hulls of $\mathcal{F}$ and $\mathcal{H}$, producing approximative polyhedra representing upper arm and forearm respectively. For the 3-DOF model, the same technique is used but only for one link.

4.6.6 Interval Transformation Matrices

This method is presented by the authors in [79] for use in predicting the occupancy of serial-link robots. Let $\mathcal{R}_q$ be the projection of the state space reachable set onto the joint space. We over-approximate the occupancy from $\mathcal{R}_q$.

The entire set of possible transformation matrices of the coordinate system of the i^{th} joint $\{F(q)^{(i)}|q \in \mathcal{R}_q\}$ is enclosed in an interval matrix, which is then applied to the i^{th} limb segments, i.e. we find the matrix of intervals $\mathcal{M}^{(i)}$, where:

$$\mathcal{M}^{(i)} \supseteq \{F(q)^{(i)} \in \mathbb{R}^{4\times 4}|q \in \mathcal{R}_q\}$$

The occupancies of each link in their own coordinate spaces are calculated offline, and enclosed in *zonotopes* $\mathcal{Z}_{link,i}$, where i refers to the i^{th} link.

We then transform each link $\mathcal{Z}_{link,i}$ by $\mathcal{M}^{(i)}$ to obtain an over-approximation of the occupancy of the link, see Fig. 41. This method is used for serial link robots where the shape of the links is well known and the reachable sets are small.

The multiplication of a matrix of intervals with a zonotope is described in [68] and results in an over-approximative zonotope. This method therefore represents each link as an over-approximative zonotope enclosing its occupancy:

$$\Gamma(t) = \bigcup_{i=1}^{4} \mathcal{M}^{(i)} \mathcal{Z}_{link,i}$$

In the next section we compare the novel representation method presented in Sec. 4.6 with the methods above, for both 3-DOF and 4-DOF models, on real data, in order to validate the models and to justify simplifications made.

4.7 Validation

We test our approach on two persons from our trials, one male and one female, executing movements other than those used to determine the parameters of the model. The movements executed were catching a football and catching a tennis ball, and are chosen because the subject could not "overplan" the movement before execution, i.e. the movements were close to involuntary. This is expected to be similar to human behaviour in an environment where humans are comfortable working alongside robots. The proposed approach is compared with

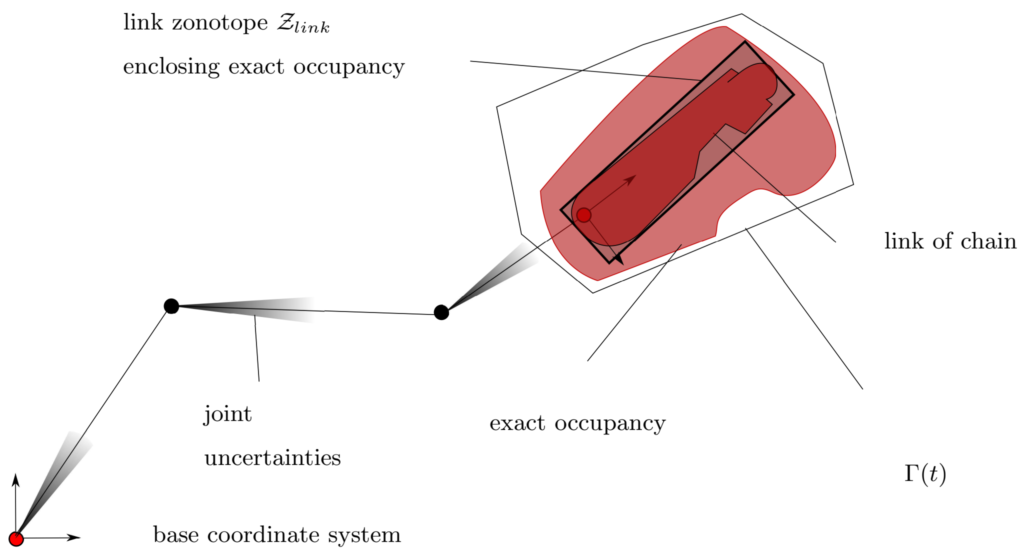


Figure 41: Uncertainties in joint positions lead to uncertainty in the occupancy of the link, which can be over-approximated by a zonotope

the approaches presented in Sec. 4.6.6 and 4.6.5. The movements are sampled at intervals of t_{final} and it is checked whether $\Gamma(t_{final})$ encloses the marker positions in interval $k+1$ given the marker positions from the interval k. The qualities compared are:

1. volume as fraction of "exact" volume,
2. computation time as fraction of prediction time t_{final},
3. percentage of time steps for which all markers at t_{final} were included in the reachable occupancy $\Gamma(t_{final})$

For the over-approximative methods, the minimum and maximum length of the upper arm cylinder were taken to be 0.3m and 0.4m respectively, and its radius was 0.075m. The maximum length of the forearm was 0.5m and its radius was 0.1m. Following from Eq. (8), in the 3-DOF model, the radius of the cylinder was set at 0.25m, half the forearm length. As the third criterion checks containment of the marker positions, in the sampling approach in 4-DOF, the occupancy samples of the links are set as the marker positions as illustrated in Fig. 19: on the upper arm, RSHO, RUPA and RELB; on the forearm, RELB, RFRM, RWRA, RWRB and RFIN (and the respective markers on the left arm). In the sampling approach in 3-DOF, 12 samples were taken from the edge of the cylindrical link, constituting a hexagonal prism. The "exact" volume was taken to be the volume using the sampling method with 256 samples in the reachable set for the 4-DOF model and 64 samples for the 3-DOF model. In contrast, the sampling method, was implemented with samples only at the corners of the zonotope, i.e. two samples per generator. The sensor error was set at $\pm 0.05 rad$ for the first two joints, $\pm 0.1 rad$ for the 3rd and 4th rotational joints and $\pm 20 mm$ for the prismatic joint.

For each subject and each movement, the left and right arm occupancies were tested and averaged, for values of t_{final} of 12.5ms and 20.8ms. For $t_{final} > 20.8$, $\Gamma(t_{final})$ becomes very large, although this increase is less severe in the 3DOF model. This is illustrated in Fig. 42.

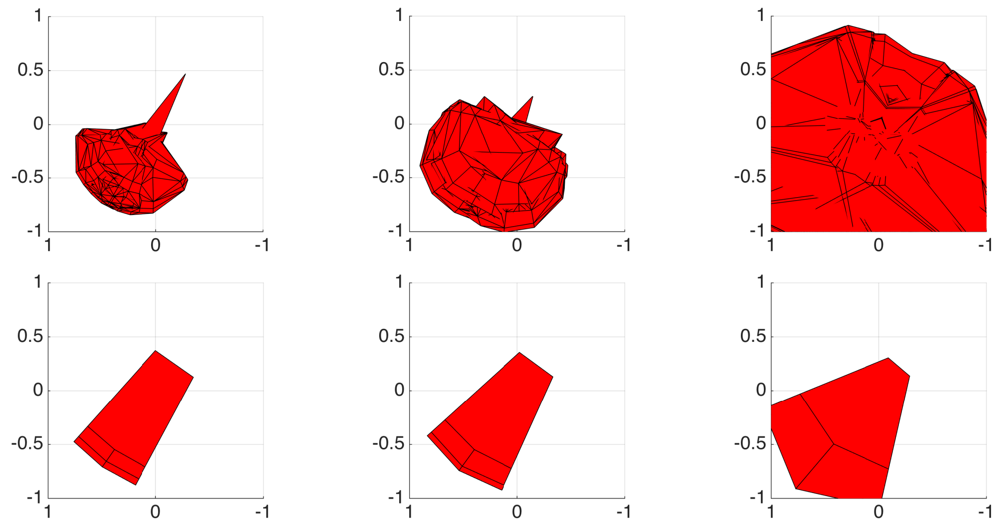


Figure 42: Left to right: reachable occupancies at 12.5ms, 20.8ms and 37.5ms respectively, for 4-DOF (top) and 3-DOF (bottom) models. Scale in meters; origin at shoulder.

Computations were performed on MATLAB R2014b running on a Macbook Pro with OSX 10.9.5. For the reachability analysis, the toolbox CORA developed in UnCoVerCPS was used, which uses INTLAB [80] for interval computations. To generate the robot kinematics and dynamics, the Robotic Toolbox [81] was used. For the representation as polytopes, the MPT toolbox [82] was used, which uses YALMIP [83] for linear matrix inequality computations.

Table 6: 4-DOF Model, $12.5 ms$

Approach:	Sampling	Interval Matrices	Polytopes
Volume ratio	0.75	5,116	3.92
Computation time	5.73	0.40	81.25
Entire arm contained	33.4%	100%	100%

Table 7: 3-DOF Model, 12.5ms

Approach:	Sampling	Interval Matrices	Polytopes
Volume ratio	0.99	5.44	1.51
Computation time	4.13	0.25	0.79
Entire arm contained	99.8%	100%	100%

Table 8: 4-DOF Model, 20.8ms

Approach:	Sampling	Interval Matrices	Polytopes
Volume ratio	0.54	1.05×10^5	3.58
Computation time	3.22	0.31	82.21
Entire arm contained	36.5%	100%	100%

Table 9: 3-DOF Model, 20.8ms

Approach:	Sampling	Interval Matrices	Polytopes
Volume ratio	0.99	12.06	1.34
Computation time	2.43	0.19	0.52
Entire arm contained	100%	100%	100%

In both the 3-DOF and 4-DOF model, the clear advantage of the interval matrix method in terms of computation time is offset by the fact that the reachable set quickly grows to fill the entire arm workspace and beyond. On the other hand, the sampling method in 4-DOF has the lowest volume but does not enclose all the points in the arm; its computation time, furthermore, is unsatisfactory in both models. In contrast, our method encloses the entire arm and the size of the over-approximation is less than the volume of the reachable occupancy.

However, the computation time of the 3-DOF model is clearly superior to that of the 4-DOF model. The polytope resulting from the 3-DOF model is defined by fewer halfspaces, so it will also be faster in collision-checking algorithms.

Volume ratio is also not a perfect measure of the quality of enclosure: a long, thin, or spiky over-approximation is likely to give more false positives in collision-checking algorithms than a sphere-like occupancy. Offline computation could also contribute to a tighter-bounded reachable occupancy.

The fact that the reachable set of the 4-DOF model grows so fast can be explained by the large accelerations possible in joints 3 and 4: $|\ddot{q}_3|_{\max} = 7900 rad/s^2$ and $|\ddot{q}_4|_{\max} = 1800 rad/s^2$. The 3-DOF model, in incorporating movements of the forearm into a prismatic joint, avoids such problems.

4.8 Conclusion on Human Arm Prediction

We address the problem of tightly bounding reachable occupancies of the human arm for a range of parameters and all possible movements. We present a novel method for bounding the occupancy of the human arm online from sensor data. Our approach fits motion capture data to a kinematic model to generate the uncertain system dynamics. In real time, it fits sensor data to the kinematic model, uses the dynamics generated offline to generate reachable sets of kinematic model and then converts this to a reachable occupancy of the arm in Cartesian space. The model is validated by checking against movements not used in the generation of the dynamic model. The reachable occupancy is represented by convex polyhedra for ease of interface with a collision checker, and can be used in path planning algorithms to verify collision free robot trajectories where humans and robots share a collaborative workspace. Future work should focus on reliable state estimation techniques from sensors that can be employed in HRI scenarios and implementing a safe path planning algorithm to ensure the entire algorithm stays within real-time.

5 Conclusion

We present techniques for predicting the behavior of surrounding entities. In the automated driving setting, those entities are surrounding traffic participants, while in the human-robot collaboration setting, surrounding entities are humans surrounding the robot. Since the exact behavior of traffic participants or humans is not exactly known, we add set-based uncertainty. Due to set-based uncertainties, we use reachability analysis to predict the set of all possible behaviors in an over-approximative way. The proposed abstraction of system dynamics significantly reduces computation time and allows us to use the presented techniques online during the operation of the ego system, i.e., the system to be verified.

Based on alternating measurements and reachability analysis, the set of solutions is propagated forward in time until new measurements are available for the next iteration. After each measurement, the measured states are enlarged by the set of possible measurement errors. The predicted occupancies over time are used as a forbidden region for the control design, providing the constraints of the controller.

Although the approaches for predicting traffic participants and humans differ in details, the main ideas are similar and are implemented with the support of new reachability analysis techniques developed in this work. The prediction of surrounding entities will be integrated into the automated vehicles use case and the robotic use case.

References

[1] G. Lafferriere, G. J. Pappas, and S. Yovine, "Symbolic reachability computation for families of linear vector fields," *Symbolic Computation*, vol. 32, pp. 231–253, 2001.

[2] M. Althoff and B. H. Krogh, "Zonotope bundles for the efficient computation of reachable sets," in *Proc. of the 50th IEEE Conference on Decision and Control*, pp. 6814–6821, 2011.

[3] "European Commission. Mobility and Transport." http://ec.europa.eu/transport /road_safety/index en.htm.

[4] K. D. Kusano and H. C. Gabler, "Safety benefits of forward collision warning, brake assist, and autonomous braking systems in rear-end collisions.," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1546–1555, 2012.

[5] A. Barth and U. Franke, "Where Will the Oncoming Vehicle be the Next Second?," in *Proceedings of the IEEE Intelligent Vehicles Symposium*, (Eindhoven), pp. 1068–1073, Springer, June 2008.

[6] A. Eidehall, "Multi-target threat assessment for automotive applications," in *Proc. of the 14th Int. IEEE Conference on Intelligent Transportation Systems*, pp. 433–438, 2011.

[7] J.-H. Kim and D.-S. Kum, "Threat prediction algorithm based on local path candidates and surrounding vehicle trajectory predictions for automated driving vehicles," *Intelligent Vehicles Symposium (IV)*, pp. 1220 – 1225, June-July 2015.

[8] M. Brännström, E. Coelingh, and J. Sjöberg, "Model-based threat assessment for avoiding arbitrary vehicle collisions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 658–669, 2010.

[9] J. Wei, J. M. Snider, T. Gu, J. M. Dolan, and B. Litkouhi, "A behavioral planning framework for autonomous driving," in *Proceedings of the 2014 IEEE Intelligent Vehicles Symposium*, June 2014.

[10] N. Kaempchen, B. Schiele, and K. C. J. Dietmayer, "Situation assessment of an autonomous emergency brake for arbitrary vehicle-to-vehicle collision scenarios.," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 4, pp. 678–687, 2009.

[11] A. Eidehall and L. Petersson, "Statistical threat assessment for general road scenes using Monte Carlo sampling," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, pp. 137–147, 2008.

[12] A. E. Broadhurst, S. Baker, and T. Kanade, "Monte Carlo road safety reasoning," in *Proc. of the IEEE Intelligent Vehicles Symposium*, pp. 319–324, 2005.

[13] T. Gindele, S. Brechtel, and R. Dillmann, "Learning context sensitive behavior models from observations for predicting traffic situations," *IEEE International Conference on Intelligent Transportation Systems*, pp. 1764 – 1771, 2013.

[14] D. Petrich, T. Dang, D. Kasper, G. Breuel, and C. Stiller, "Map-based long term motion prediction for vehicles in traffic environments," *IEEE International Conference on Intelligent Transportation Systems*, pp. 2166 – 2172, 2013.

[15] M. Liebner, C. Ruhhammer, F. Klanner, and C. Stiller, "Generic driver intent inference based on parametric models," *IEEE International Conference on Intelligent Transportation Systems*, pp. 268 – 275, 2013.

[16] A. Lambert, D. Gruyer, G. S. Pierre, and A. N. Ndjeng, "Collision probability assessment for speed control," in *Proc. of the 11th International IEEE Conference on Intelligent Transportation Systems*, pp. 1043–1048, 2008.

[17] M. Althoff, O. Stursberg, and M. Buss, "Model-Based Probabilistic Collision Detection in Autonomous Driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 2, pp. 299–310, 2009.

[18] M. Althoff, O. Stursberg, and M. Buss, “Safety assessment of driving behavior in multi-lane traffic for autonomous vehicles,” in *Proc. of the IEEE Intelligent Vehicles Symposium*, pp. 893–900, 2009.

[19] D. Hess, M. Althoff, and T. Sattel, “Formal verification of maneuver automata for parameterized motion primitives,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, September 14-18, 2014*, pp. 1474–1481, 2014.

[20] J. van den Berg, D. Ferguson, and J. Kuffner, “Anytime path planning and replanning in dynamic environments,” in *Proc. of the International Conference on Robotics and Automation*, pp. 2366–2371, 2006.

[21] C. Schmidt, F. Oechsle, and W. Branz, “Research on trajectory planning in emergency situations with multiple objects,” in *Proc. of the IEEE Intelligent Transportation Systems Conference*, pp. 988–992, 2006.

[22] L. E. Dubins, “On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents,” *American Journal of Mathematics*, vol. 79, pp. 497–516, 1957.

[23] C. F. Chung, T. Furukawa, and A. Ǵoktogan, “Coordinated Control for Capturing a Highly Maneuverable Evader using Forward Reachable Sets.,” in *ICRA*, pp. 1336–1341, IEEE, 2006.

[24] K. Kim and P. R. Kumar, “An MPC-Based Approach to Provable System-Wide Safety and Liveness of Autonomous Ground Traffic,” *IEEE Trans. Automat. Contr.*, vol. 59, no. 12, pp. 3341–3356, 2014.

[25] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, “Predictive Active Steering Control for Autonomous Vehicle Systems,” *IEEE Trans. Contr. Sys. Techn.*, vol. 15, no. 3, pp. 566–580, 2007.

[26] J. Wang, J. Wu, and Y. Li, “The Driving Safety Field Based on Driver–Vehicle–Road Interactions,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, pp. 2203 – 2214, August 2015.

[27] B. Vanholme, D. Gruyer, B. Lusetti, S. Glaser, and S. Mammar, “Highly automated driving on highways based on legal safety,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 333–347, 2013.

[28] P. Bender, J. Ziegler, and C. Stiller, "Lanelets: Efficient map representation for autonomous driving.," in *Intelligent Vehicles Symposium*, pp. 420–425, 2014.

[29] I. Shim, J. Choi, S. Shin, T.-H. Oh, U. Lee, B. Ahn, D.-G. Choi, D. H. Shim, and I. S. Kweon, "An autonomous driving system for unknown environments using a unified map," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, pp. 1999 – 2013, 2015.

[30] C. Guo, J. Meguro, Y. Kojima, and T. Naito, "A multimodal ADAS system for unmarked urban scenarios based on road context understanding," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 4, pp. 1690–1704, 2015.

[31] M. Althoff and J. M. Dolan, "Online verification of automated road vehicles using reachability analysis," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, 2014.

[32] L. M. Surhone, M. T. Timpledon, and S. F. Marseken, eds., *Vienna Convention on Road Traffic*. VDM Publishing, 2010.

[33] M. Althoff, D. Heß, and F. Gambert, "Road occupancy prediction of traffic participants," in *Proc. of the 16th International IEEE Conference on Intelligent Transportation Systems*, pp. 99–105, 2013.

[34] M. K. Agoston, *Computer Graphics and Geometric Modeling. Implementation and Algorithms*. Springer, 2004.

[35] K. H. G. Greiner, "Efficient clipping of arbitrary polygons," *ACM Transactions on Graphics*, pp. 71–83, 1998.

[36] "OpenSteetMap." www.openstreetmap.org.

[37] "JavaOpenSteetMap." https://josm.openstreetmap.de.

[38] S. Albrecht, M. Leibold, and M. Ulbrich, "A bilevel optimization approach to obtain optimal cost functions for human arm movements," *Numerical Algebra, Control and Optimization*, vol. 2, no. 1, pp. 105–127, 2012.

[39] P. Morasso, "Three dimensional arm trajectories," *Biological Cybernetics*, vol. 48, no. 3, pp. 187–194, 1983.

[40] T. Flash and N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model," *The Journal of Neuroscience*, vol. 5, no. 7, pp. 1688–1703, 1985.

[41] V. De Sapio, J. Warren, and O. Khatib, “Predicting reaching postures using a kinematically constrained shoulder model,” in *Advances in Robot Kinematics* (J. Lenarčič and B. Roth, eds.), pp. 209–218, Springer Netherlands, 2006.

[42] T. Pozzo, J. McIntyre, G. Cheron, and C. Papaxanthis, “Hand trajectory formation during whole body reaching movements in man,” *Neuroscience Letters*, vol. 240, no. 3, pp. 159 – 162, 1998.

[43] S. Albrecht, C. Passenberg, M. Sobotka, A. Peer, M. Buss, and M. Ulbrich, “Optimization criteria for human trajectory formation in dynamic virtual environments,” in *Haptics: Generating and Perceiving Tangible Sensations* (A. Kappers, J. van Erp, W. Bergmann Tiest, and F. van der Helm, eds.), vol. 6192 of *Lecture Notes in Computer Science*, pp. 257–262, Springer Berlin Heidelberg, 2010.

[44] D. Bortot, H. Ding, A. Antonopolous, and K. Bengler, “Human motion behavior while interacting with an industrial robot,” *Work*, vol. 41, no. 1, pp. 1699–1707, 2012.

[45] F. Rohrmüller, M. Althoff, D. Wollherr, and M. Buss, “Probabilistic mapping of dynamic obstacles using Markov chains for replanning in dynamic environments,” in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2504–2510, 2008.

[46] H. Ding, K. Wijaya, G. Reißig, and O. Stursberg, “Online computation of safety-relevant regions for human robot interaction,” in *Proc. 43rd Intl. Symp. Robotics (ISR), Taipei, Taiwan, Aug. 29-31, 2012*, 2012.

[47] R. Vatcha and J. Xiao, “Detection of robustly collision-free trajectories in unpredictable environments in real-time,” *Autonomous Robots*, vol. 37, no. 1, pp. 81–96, 2014.

[48] D. Kulić and E. Croft, “Real-time safety for human - robot interaction,” in *Proc. of 12th International Conference on Advanced Robotics, 2005*, pp. 719–724, July 2005.

[49] M. Polverini, A. Zanchettin, and P. Rocco, “Real-time collision avoidance in human-robot interaction based on kinetostatic safety field,” in *Proc. of 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4136–4141, Sept 2014.

[50] A. Zanchettin, N. Ceriani, P. Rocco, H. Ding, and B. Matthias, “Safety in human-robot collaborative manufacturing environments: Metrics and control,” *IEEE Transactions on Automation Science and Engineering*, vol. PP, no. 99, pp. 1–12, 2015.

[51] K. Holzbaur, W. Murray, and S. Delp, "A model of the upper extremity for simulating musculoskeletal surgery and analyzing neuromuscular control," *Annals of Biomedical Engineering*, vol. 33, no. 6, pp. 829–840, 2005.

[52] M. Damsgaard, J. Rasmussen, S. T. Christensen, E. Surma, and M. de Zee, "Analysis of musculoskeletal systems in the anybody modeling system," *Simulation Modelling Practice and Theory*, vol. 14, no. 8, pp. 1100 – 1111, 2006.

[53] E. Demircan, T. Besier, S. Menon, and O. Khatib, "Human motion reconstruction and synthesis of human skills," in *Advances in Robot Kinematics: Motion in Man and Machine* (J. Lenarcic and M. M. Stanisic, eds.), pp. 283–292, Springer Netherlands, 2010.

[54] E. Demircan, T. Besier, and O. Khatib, "Muscle force transmission to operational space accelerations during elite golf swings," in *2012 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1464–1469, May 2012.

[55] E. Asarin, T. Dang, G. Frehse, A. Girard, C. Le Guernic, and O. Maler, "Recent progress in continuous and hybrid reachability analysis," in *Proc. of the 2006 IEEE Conference on Computer Aided Control Systems Design*, pp. 1582–1587, 2006.

[56] A. Girard and G. J. Pappas, "Verification using simulation," in *Hybrid Systems: Computation & Control*, LNCS 3927, pp. 272–286, Springer, 2006.

[57] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent Hamilton–Jacobi formulation of reachable sets for continuous dynamic games," *IEEE Trans. on Automatic Control*, vol. 50, pp. 947–957, 2005.

[58] A. Chutinan and B. H. Krogh, "Computational techniques for hybrid system verification," *IEEE Trans. on Automatic Control*, vol. 48, no. 1, pp. 64–75, 2003.

[59] A. Girard, C. Le Guernic, and O. Maler, "Efficient computation of reachable sets of linear time-invariant systems with inputs," in *Hybrid Systems: Computation & Control*, LNCS 3927, pp. 257–271, Springer, 2006.

[60] A. B. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis," in *Hybrid Systems: Computation and Control*, LNCS 1790, pp. 202–214, Springer, 2000.

[61] A. Girard and C. Le Guernic, "Efficient reachability analysis for linear systems using support functions," in *Proc. of the 17th IFAC World Congress*, pp. 8966–8971, 2008.

[62] O. Stursberg and B. H. Krogh, “Efficient representation and computation of reachable sets for hybrid systems,” in *Hybrid Systems: Computation and Control*, LNCS 2623, pp. 482–497, Springer, 2003.

[63] A. A. Amis, D. Dowson, and V. Wright, “Analysis of elbow forces due to high-speed forearm movements,” *Journal of Biomechanics*, vol. 13, pp. 825–831, 2015/05/14 1980.

[64] N. Klopčar, M. Tomšič, and J. Lenarčič, “A kinematic model of the shoulder complex to evaluate the arm-reachable workspace,” *Journal of Biomechanics*, vol. 40, pp. 86–91, 2015/05/14 2001.

[65] C. Högfors, B. Peterson, G. Sigholm, and P. Herberts, “Biomechanical model of the human shoulder joint—ii. the shoulder rhythm,” *Journal of Biomechanics*, vol. 24, pp. 699–709, 2015/06/28 1991.

[66] J. C. Otis, C. C. Jiang, T. L. Wickiewicz, M. G. Peterson, R. F. Warren, and T. J. Santner, “Changes in the moment arms of the rotator cuff and deltoid muscles with abduction and rotation.,” *The Journal of Bone & Joint Surgery*, vol. 76, no. 5, pp. 667–676, 1994.

[67] M. Althoff, O. Stursberg, and M. Buss, “Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization,” in *Proc. of the 47th IEEE Conference on Decision and Control*, pp. 4042–4048, 2008.

[68] M. Althoff and B. H. Krogh, “Reachability analysis of nonlinear differential-algebraic systems,” *IEEE Trans. on Automatic Control*, vol. 59, no. 2, pp. 371–383, 2014.

[69] M. Althoff, *Reachability Analysis and its Application to the Safety Assessment of Autonomous Cars.* Dissertation, Technische Universität München, 2010.

[70] D. Hahn, M. Olvermann, J. Richtberg, W. Seiberl, and A. Schwirtz, “Knee and ankle joint torque-angle relationships of multi-joint leg extension,” *Journal of Biomechanics*, vol. 44, pp. 2059–2065, July 2011.

[71] B. M. Nigg and W. Herzog, *Biomechanics of the Musculo-skeletal System.* Wiley, 1999.

[72] K. B. F. Guenzkofer, H. Bubb, “The validity of maximum force predictions based on single-joint torque measurements,” in *The 2nd International Digital Human Modeling Symposium*, 2013.

[73] B. Grünbaum, *Convex Polytopes.* Springer, 2 ed., 2003.

[74] I. Todhunter and J. G. Leathem, *Spherical Trigonometry.* Macmillan, 1901.

[75] A. D. Alexandrov, *Convex Polyhedra.* Springer, 2005.

[76] H. S. M. Coxeter and S. L. Greitzer, *Geometry Revisited.* Mathematical Association of America, 1967.

[77] W. Lietzmann, *Elementare Kugelgeometrie.* Vandenhoeck & Ruprecht, 1949.

[78] T. McMahon, S. Thomas, and N. Amato, "Sampling-based motion planning with reachable volumes: Theoretical foundations," in *2014 IEEE International Conference on Robotics and Automation*, pp. 6514–6521, May 2014.

[79] A. Pereira and M. Althoff, "Safety control of robots under computed torque control using reachable sets," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2015.

[80] S. Rump, "INTLAB - INTerval LABoratory," in *Developments in Reliable Computing* (T. Csendes, ed.), pp. 77–104, Dordrecht: Kluwer Academic Publishers, 1999. `http://www.ti3.tuhh.de/rump/`.

[81] P. I. Corke, *Robotics, Vision & Control: Fundamental Algorithms in Matlab.* Springer, 2011.

[82] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *Proc. of the European Control Conference*, (Zürich, Switzerland), pp. 502–510, July 17–19 2013. `http://control.ee.ethz.ch/~mpt`.

[83] J. Löfberg, "Yalmip : a toolbox for modeling and optimization in matlab," in *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*, pp. 284–289, Sept 2004.