Detecting Avoidance Behaviors Between Moving Object Trajectories

Blind Review
(*September, 2014*)

Several algorithms have been proposed in the last few years for mining different mobility patterns from trajectories, such as flocks, chasing, meeting, and convergence. An interesting behavior that has not been much explored in trajectory pattern mining is *avoidance*. In this paper we define the avoidance behavior between moving object trajectories, providing a set of theoretical definitions to precisely describe various kinds of avoidance, and propose an effective algorithm for detecting avoidances. The proposed method is quantitatively evaluated on a real-world dataset, and correctly detects with high precision the quasi totality of the trajectory pairs that exhibit avoidance behaviors (F-measure up to 95%).

Keywords: Trajectory Avoidance Detection; Spatio-Temporal Analysis; Trajectory mining
.

1. Introduction and Motivation

Current advances in mobile technology such as GPS and smartphones have increased the interest in mobility data analysis in several application domains such as security, smart cities, transportation systems, urban planning, and biological studies. As a consequence, several algorithms have been proposed for discovering various types of behaviors in trajectory data such as the T-patterns (Giannotti *et al.* 2007), flocks (Laube *et al.* 2005, Wachowicz *et al.* 2011), meet (Gudmundsson and van Kreveld 2006), periodic movements (Li *et al.* 2010, Trasarti *et al.* 2011), anomalous traffic patterns (Pang *et al.* 2013), chasing (de Lucca Siqueira and Bogorny 2011), etc. In (Dodge *et al.* 2008), a taxonomy with different types of trajectory behaviors is proposed, while a summary of the most well known trajectory behaviors (also called patterns) is presented in (Parent *et al.* 2013).

In this paper we address the problem of trajectory *avoidance detection*. Here we distinguish two main classes of works about avoidance: *collision avoidance* and *avoidance detection*. Collision avoidance is a well studied and established field with the main objective to suggest a new route (trajectory) to avoid collision. There is a vast literature on this topic in areas such as robotics, vehicle simulation systems and vehicle embedded systems, as detailed in the Related Works section. This paper focuses on the second class of avoidance: the avoidance detection. In this domain the objective is to detect if a moving object has avoided a static object (area), as shown in Figure 1(a), and a moving object avoiding another one, as shown in Figure 1(b, c, and d).

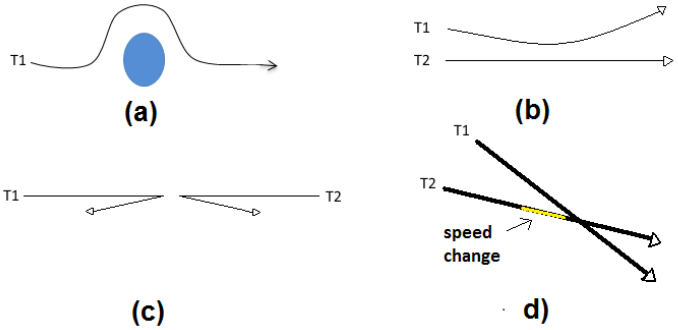


Figure 1. Different kinds of avoidance behaviors: avoidance with respect to a static object (a), and avoidance between moving objects: *individual* (b), *mutual* (c), and *individual* induced by a change in speed (d).

Static avoidance detection can be interesting for discovering suspicious behaviors as, for instance, objects avoiding a surveillance camera, a police patrol, or speed controllers. A first treatment of this type of avoidance is proposed in (Alvares *et al.* 2011).

Avoidance detection between moving objects is useful in several application domains. For example, in security applications it may reveal suspicious behaviors among people, such as criminals or terrorists that avoid policemen. In marine surveillance, ships with illicit products or illegal immigrants may avoid cost guard boats. In computer games, the avoidance behavior can be useful to detect, for instance, the avoided enemies, while in soccer games it may be useful to analyze players avoiding markers. In zoological studies, avoidance detection may reveal how preys avoid predators (e.g., at which distance, by changing direction or changing speed). The discovery of this type of avoidance, however, is more challenging than the avoidance of static objects, and the first questions that raise are: what are the main features that characterize an avoidance between two trajectories? Who is avoiding who? At what distance two objects initiate an avoidance?

In this paper we introduce a new method for trajectory avoidance detection, formalizing the concept of trajectory avoidance behavior, to identify every instance of this behavioral pattern in historical movement traces. Figure 1 shows three examples of avoidance behavior addressed in this paper: Figure 1(b) shows an example of trajectory avoidance where T_1 avoids trajectory T_2 by changing its direction. Figure 1(c) shows a mutual avoidance behavior, where both T_1 and T_2 avoid each other by changing their direction. In Figure 1(d), although both trajectories have a spatial intersection relationship, T_2 avoids T_1 by slowing down the speed in order not to spatio-temporally intersect T_1.

This specific problem has not received much attention in the literature. Li in (Li *et al.* 2013) has made the first attempt to address trajectory avoidance detection looking for attraction and avoidance relationships between pairs of trajectories. In general terms, this work considers the frequency of meetings to define and avoidance. When a pair of objects frequently moves close to each other, an attraction relationship is characterized, and when the pair rarely moves close to each other, and avoidance is detected. As a result, this method measures the degree of attraction or avoidance between two trajectories.

We claim that an avoidance *between trajectories*, as the examples shown in Figure 1 is characterized by a change of movement behavior. Therefore, we define an avoidance behavior making a forecast of the possible movement of two trajectories and compare this forecast with the real movement, in order to detect a change of behavior. In summary, we make the following contributions in this paper: (i) we introduce a framework which defines the avoidance between pairs of trajectories considering changes of behavior in

speed or direction, so to cover the three types of avoidance shown in Figure 1(b, c, d); (ii) we present an algorithm which is able to automatically detect every avoidance between two trajectories, and to work on real-world trajectories collected at different sampling rates; (iii) we propose a criteria to classify any avoidance as *weak*, *mutual* or *individual*, on the basis of factual evidence related to changes in behavior of the involved trajectories; and (iv) we present a fusion detector for analyzing the avoidance with different sets of parameters and fuse the results in a unique output.

The rest of the paper is organized as follows: Section 2 presents the related work. Section 3 introduces some basic definitions. Section 4 illustrates the new definitions for avoidance detection while Section 5 proposes an algorithm to compute avoidance behavior. Section 6 describes experiments on real trajectories, while Section 7 concludes the paper and suggests directions of future research.

2. Related Work

As mentioned before, we can distinguish two main classes of works about avoidance: *collision avoidance* and *avoidance detection*. Collision avoidance deals with models, systems, and practices designed to prevent vehicles such as cars, ships, and airplanes from colliding with other vehicles. Therefore, the focus relies on detecting a future collision, and change (or suggest a change) of the current route of one or more of the involved vehicles to avoid a collision.

There is a vast literature in these domains for collision avoidance. Just a few examples of works on collision avoidance in vehicle systems include: for cars (Kim *et al.* 2007), (Xausa *et al.* 2012), (Hafner *et al.* 2013), (Nedevschi *et al.* 2009), for ships (Liu and Shi 2005),(Mou *et al.* 2010), and for aircrafts (Shandy and Valasek 2001),(Richards and How 2002) and (Pechoucek and Sislak 2009).

In (Kim *et al.* 2007) the proposal is to minimize the safety distance error and to regulate the relative speed between two vehicles, so to avoid rear-end collision, using hierarchical longitudinal control. Xausa in (Xausa *et al.* 2012) proposes a method for computing a car trajectory towards a safe final state, as soon as an obstacle is detected by a sensor (e.g. radar or lidar). Two scenarios are considered: a car that is overtaking a stationary object and a car that is overtaking when a car is approaching from the opposite direction. The solution is based on the car model with two control variables (steering velocity and braking force), state variables (speed, yaw angle, yaw angle rate, the center of gravity and direction) and a state dynamics defined by a system of differential equations. The work is done in a simulation basis. Hafner in (Hafner *et al.* 2013) presents experimental results for an active control intersection collision avoidance system implemented on modified Lexus test vehicles. The system utilizes vehicle-to-vehicle dedicated short-range communications to share safety critical state information. Safety is achieved in potential collision scenarios by controlling the velocities of both vehicles with automatic brake and throttle commands.

Another approach for car collision avoidance considers pedestrians (Nedevschi *et al.* 2009), where the use of stereo cameras on board of vehicles supports the detection of pedestrians with the aim of avoiding collisions between cars and pedestrians.

In the domain of ships, for instance, Liu (Liu and Shi 2005) proposed a fuzzy-neural inference network that learns a set of examples from a set of rules defined by the International Regulations for Preventing Collisions at Sea. Based on the learned examples, the method suggests direction changes of the ship to avoid a possible collision. The main input data are the ships direction and speed, the distance between them, and the type of

water area (sea, coast, limit water). The model only considers cases where an encounter situation is already detected. The output is the set of actions to avoid the collision.

In the aircraft domain, for instance, Richards (Richards and How 2002) considers a set of aircrafts where each one has a known destination, and its trajectory will be a strait line from the current point to its destination. The speed of the aircrafts is known and constant, and it is assumed that the aircrafts fly in layers. The contribution of the paper is a linear model to modify the aircraft routes in order to avoid a collision when two or more aircrafts become close to each other. The model considers real dynamics constraints to be more realistic.

Another domain where collision-avoidance is well studied is in robotics. In this domain, when the robot is calculating its route (its possible trajectory) it should consider the known obstacles an avoid them. After, when the robot is moving, following the planned route, if it detects an obstacle it should deviate the obstacle executing a collision-avoidance. Some works include (Khatib 1986), (Borenstein and Koren 1991),(Khansari-Zadeh and Billard 2012), (Sun *et al.* 2014). Khansari-Zadeh in (Khansari-Zadeh and Billard 2012), for instance, uses a dynamical system-based approach to deviate the robot from the obstacles. Sun in (Sun *et al.* 2014) proposed a behaviour-based multi-robot collision avoidance to efficiently coordinate the simultaneous navigation of large robot teams. The proposed solution is based on the subsumption architecture (Brooks 1986) with simple behaviours such as FollowingWayPoint, Avoid, WaitKeepDistance and Dock, each one represented by a finite-state automaton.

In summary, the main goal of the previous works on collision-avoidance is to define a set of actions to prevent a collision for a specific moving object type (e.g. car, airplane, robot). These works take into account both the physical properties and the type of the moving object.

In what concerns *avoidance detection*, which is the focus of this paper, the objective is to determine if a moving object has avoided another moving object. Note that these two classes of works are completely different. While the former intends to change the route of a moving object considering several physical properties of the moving object (e.g. mass, center of gravity, steering angle), the later aims to discover if a past trajectory has deviated from another trajectory, considering only the trajectory points.

In avoidance detection, to the best of our knowledge, there are basically two works: (Alvares *et al.* 2011) and (Li *et al.* 2013). The former discovers moving objects that avoid static objects, and does not search for avoidances between moving objects. The latter is closer to our work, in that it looks for avoidance behaviors between moving objects. Still, as mentioned before, (Li *et al.* 2013) searches for general avoidance and attracting relationships based on frequency of meetings. She proposes a statistical approach based on permutation test: for each pair of moving objects the method outputs a value ranging in the interval $[0, 1]$ which expresses a global estimate of the level of attraction or avoidance between them.

Our work has a different objective, namely identifying each single occurrence of avoidance behavior between moving objects. Moreover, we distinguish various kinds of avoidances trying to determine who is avoiding who. More precisely, an avoidance occurrence is defined as a situation in which two objects are moving towards the same area, but either one or both change behavior whenever they come close enough to be aware of each other.

3. Preliminaries

Moving objects are entities having a time variant position, uniquely determined at each time instant. A trajectory is a continuous part of the movement of an object (Renso *et al.* 2013). For the sake of simplicity, in the following we will restrict to the 2D Euclidean space, but note that the generalization to higher dimensional spaces is straightforward.

Definition 3.1 (Movement and trajectory): The *movement* of an object o is a continuous function $M_o : R^+ \rightarrow R^2$ from the real positive numbers, representing time instants, to 2D space. Given an object o and a time interval [*tBegin*, *tEnd*], a trajectory T is the restriction of the movement M_o of the object to the given time interval. The spatio-temporal position of the object at *tBegin* (resp. *tEnd*) is called the *Begin* (resp. *End*) of the trajectory.

Often, in mobility applications, trajectories are only partially known, usually at specific time instants that correspond to position update actions, called trajectory points.

Definition 3.2 (Trajectory point): A trajectory point, or trajectory sample, of a trajectory T is a tuple (x, y, t), where $T(t) = (x, y)$ is the object position at time t (called the *timestamp* of the trajectory point).

The set of known positions of a moving object during the definition interval of a trajectory is named *trajectory track*, or trajectory *sampling*.

Definition 3.3 (Trajectory track): Given a temporally ordered sequence $\langle t_1, \ldots, t_n \rangle$ of timestamps, the track of a trajectory T for the given timestamps is the temporally ordered sequence of trajectory points $\langle p_1, \ldots, p_n \rangle$, where $p_i = (x_i, y_i, t_i)$ and $(x_i, y_i) = T(t_i)$.

A finite sequence of trajectory points can be paired with a finite sequence of interpolation functions that describe, in an analytical and continuous, way the movement of the object between each pair of consecutive trajectory points in the sequence. This sequence of pairs of trajectory points and interpolation functions is named *continuous trajectory representation*.

4. Avoidance

An *avoidance* between two moving objects occurs when both are moving towards the same area at the same time, but either one or both change their behavior when they come close enough to be aware of each other. In the following, with a slight abuse of terms, we will indistinctly use the terms trajectory and object when referring to the avoidance. This is acceptable since the trajectories referring to the same object do not temporally overlap.

In order to define the avoidance concept, we first introduce the predicates *meet* and *will-meet*. The first one expresses the fact that in a certain interval two trajectories become sufficiently close to be considered in contact, whereas the second one states that the *forecast* of these trajectories, determined by some technique on the basis of some observed behavior, will lead to a contact.

Definition 4.1 (Meet): Given two trajectories T_a and T_b, a time interval $[t_1, t_2]$ and a distance threshold δ, we define the predicate $meet_\delta$ as

$$meet_\delta(T_a, T_b, [t_1, t_2]) \equiv \exists t \in [t_1, t_2]. \, dist(T_a(t), T_b(t)) < \delta$$

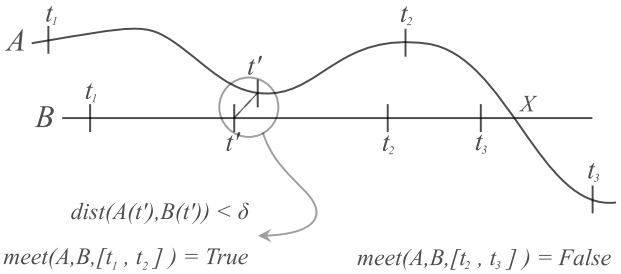


Figure 2. *A meets B* (the distance in t' is less than δ) during $[t_1, t_2]$ but not during $[t_2, t_3]$.

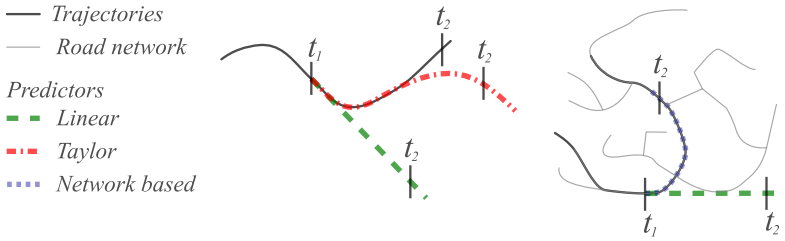


Figure 3. Different kinds of predictors based on several interpolations and on movement constraints (road network).

where $dist(p_a, p_b)$ is the Euclidean distance of the points p_a and p_b.

When the predicate is satisfied we also say that T_a *meets* T_b during $[t_1, t_2]$ with threshold δ. In case the threshold δ is evident from the context it will be omitted in the predicate notation, writing *meet* instead of $meet_\delta$.

Figure 2 illustrates a pair of trajectories, A and B, during the time interval $[t_1, t_3]$. The predicate *meet* is true for $[t_1, t_2]$, since at time t' the distance of the two trajectories is less than δ. Note that for the interval $[t_2, t_3]$, even if the two trajectories have a spatial intersection X the *meet* predicate is false since the distance between the two moving objects at any instant $t \in [t_2, t_3]$ is always greater than δ. In fact, A and B cross X at different time instants.

Having a way of predicting the movement of the trajectories on the basis of what happened in the past, we can establish if two trajectories will meet each other or not. We next introduce an abstract notion of *movement predictor* clarifying which are the expected properties.

Definition 4.2 (Movement predictor): Given a movement domain, consisting of all possible movement functions, $\mathcal{M} = \{M \mid M : R^+ \to R^2\}$, and a temporal domain R^+, a *movement predictor* is a functional

$$forecast : \mathcal{M} \times (R^+ \times R^+) \to \mathcal{M} \; such \; that$$

$$\forall M, M' \in \mathcal{M}.\; M\,|_{]0,t1]} = M'\,|_{]0,t1]} \implies forecast(M, [t1, t2]) = forecast(M', [t1, t2])$$

The functional *forecast* maps a movement function M to its forecast movement into the interval $[t_1, t_2]$ by exploiting only the behavior of M in the past interval $]0, t_1]$. It can be defined in different ways depending on the contexts. Commonly, it is based on the assumption that the behavior does not change with respect to the recent past of the trajectory. For instance, in case of objects that move freely in space, we can use the Taylor

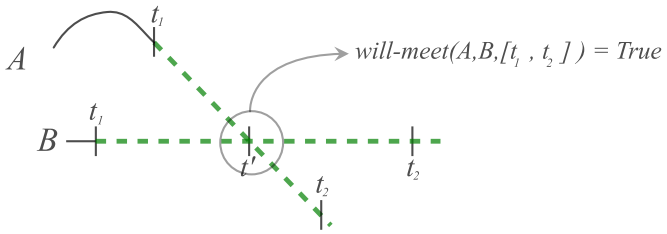


Figure 4. According to the forecasts, A *will meet* B (the distance will be less than δ) at some time t' in the time interval $[t_1, t_2]$.

series in order to estimate the next positions of the trajectory, and the more terms of these series we compute, the more precise we obtain the approximation, e.g., the first term preserves the direction, the second one the curvature. On the other hand, if the object movement is constrained by a network, we can exploit the network to predict where the object is going.

Figure 3 shows several different forecast functionals for the interval $[t_1, t_2]$. The solid line represents the actual trajectory. In Figure 3 (left and right) the dashed green line models a linear prediction, preserving the direction at time t_1; the dash-dot red line (in Figure 3(left)) shows a forecast based on a higher order Taylor series, preserving several derivatives of the trajectory; and the dotted blue line (in Figure 3 (right)) illustrates a prediction based on the knowledge of the road network.

In the rest of this paper we will use a simple linear movement predictor. We highlight, however, that this choice only affects the implementation, and any predictor could be used.

Definition 4.3 (Will-meet): Given two trajectories T_a and T_b, a time interval $[t_1, t_2]$, a movement predictor *forecast* and a threshold δ, we define the predicate *will-meet*$_\delta$ as

$$\textit{will-meet}_\delta(T_a, T_b, [t_1, t_2]) \equiv \textit{meet}_\delta(\textit{forecast}(T_a, [t_1, t_2]), \textit{forecast}(T_b, [t_1, t_2]), [t_1, t_2])$$

When the predicate is satisfied we also say that T_a is *expected to meet* T_b during $[t_1, t_2]$ with threshold δ. In case the threshold δ is evident from the context it will be omitted in the predicate notation, writing *will-meet* instead of *will-meet*$_\delta$.

Figure 4 illustrates an example where the predicate *will-meet* holds: A and B are supposed to meet at time t', provided that they maintain the same speed and direction they had at time t_1 (we used a linear movement predictor for simplicity).

With these definitions we define an avoidance between trajectories as an event that happens in a time interval where the prediction is a *meet* between two trajectories but this meet does not occur.

Definition 4.4 (Avoid): Given two trajectories T_a and T_b, a time interval $[t_1, t_2]$, and a threshold δ, we define the predicate *avoid*$_\delta$ as

$$\textit{avoid}_\delta(T_a, T_b, [t_1, t_2]) \equiv \textit{will-meet}_\delta(T_a, T_b, [t_1, t_2]) \wedge \neg \textit{meet}_\delta(T_a, T_b, [t_1, t_2])$$

When the predicate is satisfied we say that T_a and T_b *avoid* to meet, with threshold δ, during $[t_1, t_2]$. In case the threshold δ is evident from the context it will be omitted in the predicate notation, writing *avoid* instead of *avoid*$_\delta$.

The duration of the time interval $[t_1, t_2]$ is a measure of the future awareness of the moving objects (e.g., a person or an animal), that is the amount of time they are able to reliably forecast all of the involved trajectories to avoid collisions or encounters.

It is worth mentioning that although it is not explicit in Definition 4.4, this definition

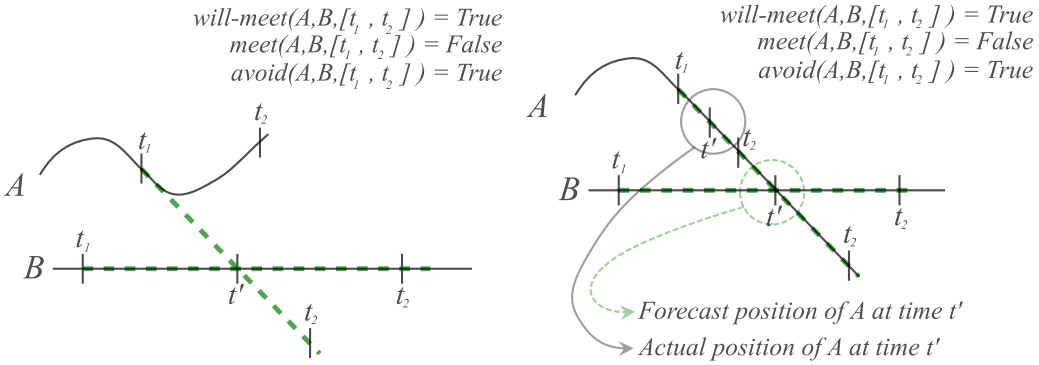


Figure 5. *A avoids B* during the time interval $[t_1, t_2]$: meet is expected according to the forecast computed at time t_1 but no actual meet happened during the given time interval.

covers the avoidance cases shown in Figure 1(b, c, and d), considering both speed and direction.

Figure 5 shows two different cases of avoidance in which trajectory B exactly fulfills the prediction (dashed green), whereas trajectory A behaves differently with respect to the forecast (dashed green): on the left, A changes direction after time t_1, and on the right it reduces its speed. As a result, the forecast of the two trajectories (A and B) are expected to meet at time $t' \in [t_1, t_2]$, but the two trajectories do not actually meet since their distance is always larger than δ (omitted for simplicity in Figure 5) during $[t_1, t_2]$. Thus, in both cases the *avoid* predicate is true.

4.1. *Avoidance Classification*

The concept of avoidance is related to a change of behavior of one or both the involved objects such that a predicted meet does not occur. We define a new predicate, *change-behavior*, to better characterize different kinds of avoidance. Informally, we regard the behavior of an object as *changed* during a time interval when there is a difference between the actual trajectory and its forecast that is sufficient to cause missed meets without any change in the other trajectory. Intuitively such a difference should have at least the same magnitude as the meet threshold δ.

Definition 4.5 (Change-behavior): Given a trajectory T, a time interval $[t_1, t_2]$, and a meet distance threshold δ, we define the predicate $\textit{change-behavior}_\delta$

$$\textit{change-behavior}_\delta(T, [t_1, t_2]) \equiv \exists t \in [t_1, t_2],\ \textit{dist}(\textit{forecast}(T, [t_1, t_2])(t), T(t)) > \delta$$

where $\textit{forecast}(T, [t_1, t_2])(t)$ and $T(t)$ are respectively the forecast and the actual position for trajectory T at time t.

When the predicate is satisfied we say that T *changes* its behavior, with threshold δ, during $[t_1, t_2]$. In case the threshold δ is evident from the context it will be omitted in the predicate notation, writing *change-behavior* instead of $\textit{change-behavior}_\delta$.

Figure 6 focuses on trajectory A of Figure 5, showing how it changes its behavior. On the left it changes direction whereas on the right it reduces the speed with respect to the forecast (green dashed line). In both cases the distance at t' between the forecast and the actual trajectory is greater than δ, hence the predicate *change-behavior* holds.

Based on the above definition, we distinguish among *weak*, *individual* and *mutual* avoid-

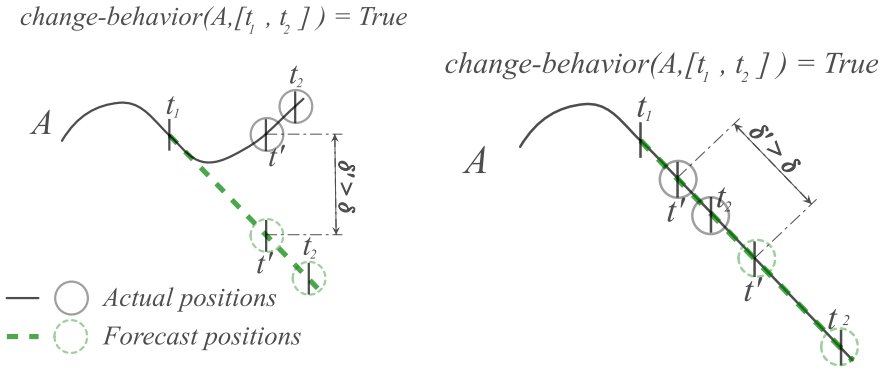


Figure 6. Trajectory A *changes behavior*: at some time t' during $[t_1, t_2]$ the distance of the actual position from the forecast position is greater than δ.

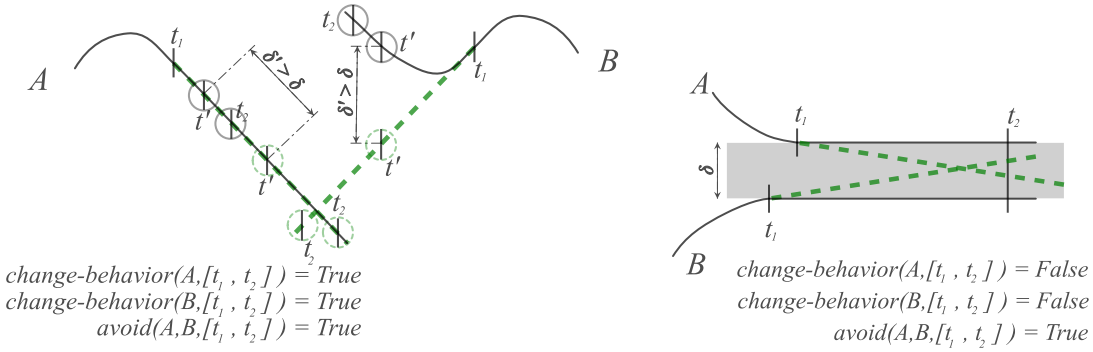


Figure 7. On the left, both A and B change behavior to avoid each other (*mutual* avoidance). On the right, according to the forecast A and B should meet but this does not happen even if both did not significantly change behavior (*weak* avoidance).

ance.

Definition 4.6 (Avoidance classification): Given two trajectories T_a and T_b, and a time interval $[t_1, t_2]$ such that $avoid_\delta(T_a, T_b, [t_1, t_2])$ is true, we classify that avoidance in the following way:

$$type\_avoid_\delta(T_a, T_b, [t_1, t_2]) = \begin{cases} mutual & \text{if } change\text{-}behavior_\delta(T_a, [t_1, t_2]) \wedge \\ & \quad change\text{-}behavior_\delta(T_b, [t_1, t_2]) \\ weak & \text{if } \neg change\text{-}behavior_\delta(T_a, [t_1, t_2]) \wedge \\ & \quad \neg change\text{-}behavior_\delta(T_b, [t_1, t_2]) \\ individual & \text{otherwise} \end{cases}$$

In other words, we have a *mutual avoidance* when there is an evident change of behavior for both trajectories, an *individual avoidance* when only one trajectory significantly changes its behavior, and a *weak avoidance* when there is an avoidance despite the fact that the behavior changes are minimal for both trajectories.

Figure 7 shows examples of *mutual* and *weak* avoidance, whereas avoidances in Figure 5 are both *individual*. We recall that trajectories are represented in solid black, forecasts in dashed green, and their specific positions at a given time are circled, respectively in solid gray and dashed green. In Figure 7 (left), A changes its behavior by reducing its speed

and this is detected since there exists a time $t' \in [t_1, t_2]$ such that the distance of the forecast position from the actual position is greater than δ. The same happens for B that changes its direction. Since both trajectories changed behavior during interval $[t_1, t_2]$, this is a *mutual* avoidance. In Figure 7 (right) at any time in $[t_1, t_2]$ for both trajectories the distance of the forecast position from the actual position is less than δ (the height of the gray rectangle). Nevertheless, trajectories were expected to meet but did not actually meet (their distance is larger than δ) and thus the *avoid* predicate is true. Since the *change-behavior* predicate is false for both trajectories, this is a *weak* avoidance.

4.2. *Problem Statement*

In this paper we address two problems of increasing complexity regarding the detection of avoidance behaviors between moving object trajectories: the *avoidance decision problem* (i.e., decide whether a pair of trajectories in a given temporal interval satisfies the predicate *avoid*) and the *avoidance search problem* (for every possible pair of trajectories find those time intervals such that the predicate *avoid* is true).

We define both problems on trajectory tracks since real trajectory datasets usually consist of sets of samplings. To obtain the continuous representation of a trajectory we use an interpolation function, namely *interp*, and we denote as $interp(\mathcal{T})$ the application of *interp* to the trajectory track $\mathcal{T}$.

We check avoidances in intervals starting at the samples of a trajectory and the duration of the interval is based on a *look-ahead* time Δt, which can be interpreted as a measure of the future awareness of moving objects. Consequently, Δt must be chosen according to their characteristics. For instance, pedestrians have a consistently smaller look-ahead time than big ships (e.g., cargos), since in the latter case, due to the tonnage and the size of the objects, changes of heading or speed are necessarily much slower. Still related to the characteristics of the moving objects is the meet threshold δ. If we consider two completely different classes of moving objects, e.g., pedestrians and ships, it is evident that the spatial extent of the actions needed to change their movements, as well as the distance at which they are considered close to another object of the same type, will be typically different and this fact can and should influence the selection of δ.

The first problem we address is the decision problem.

Definition 4.7 (Avoidance decision problem): Given two trajectory tracks $\mathcal{T}_a$,$\mathcal{T}_b$, the set of their timestamps $TS = \{timestamp(p) \mid p \in \mathcal{T}_a \vee p \in \mathcal{T}_b\}$, a time instant t, such that $t \in TS$, a look-ahead interval Δt, and a meet threshold δ, the *avoidance decision problem* consists in determining whether

$$avoid_\delta(interp(\mathcal{T}_a), interp(\mathcal{T}_b), [t, t + \Delta t]) \text{ holds.}$$

The second problem we address is a *search* problem: for each pair of trajectories we look for all the timestamps belonging to one of the two trajectory tracks such that the predicate *avoid* holds and we determine the associated type of avoidance. As a first step we formulate this problem for a pair of trajectories in order to find the set of timestamps and the relative avoidance type where an avoidance between such trajectories is detected. Then the notion will be generalized to a set of trajectories.

Definition 4.8 (Avoidances set): Given two trajectory tracks $\mathcal{T}_a$,$\mathcal{T}_b$, the set of their timestamps $TS = \{timestamp(p) \mid p \in \mathcal{T}_a \vee p \in \mathcal{T}_b\}$, a look-ahead time Δt and a meet threshold δ, we define

$$avoidances_\delta(\mathcal{T}_a, \mathcal{T}_b) = \{(t, type) \mid t \in TS \ \wedge avoid_\delta(interp(\mathcal{T}_a), interp(\mathcal{T}_b), [t, t + \Delta t]) \wedge$$

$$type\_avoid_\delta(interp(\mathcal{T}_a), interp(\mathcal{T}_b), [t, t+\Delta t]) = type\}$$

In words, a pair $(t, type) \in avoidances_\delta$ when t is a timestamp in one of the trajectory tracks and the predicate *avoid* holds for the two trajectories in the interval $[t, t+\Delta t]$. *type* is the kind of the detected avoidance.

Definition 4.9 (Avoidance search problem): Given a set of trajectory tracks $\mathcal{D}$, a look-ahead time Δt, and a meet threshold δ, the *avoidance search problem* consists in finding the set of tuples

$$\{(\mathcal{T}_a, \mathcal{T}_b, (t, type)) \mid \mathcal{T}_a \in \mathcal{D} \wedge \mathcal{T}_b \in \mathcal{D} \wedge (t, type) \in avoidances_\delta(\mathcal{T}_a, \mathcal{T}_b)\}$$

each consisting of a pair of trajectory tracks $\mathcal{T}_a, \mathcal{T}_b$, a time instant t and a type *type* such that $\mathcal{T}_a$ and $\mathcal{T}_b$ avoid to meet, with threshold δ, after t for Δt time and the kind of avoidance is specified by *type*.

In order to get a more compact representation of the result set, we next replace the set of timestamps for a couple of trajectories ($avoidances(\mathcal{T}_a, \mathcal{T}_b)$) with a disjoint set of intervals. The idea is to join two avoidances if the intervals where they are detected overlap. To this aim, we introduce the notion of *repeated avoidance* in an interval.

Definition 4.10 (Repeated avoidance): Given two trajectory tracks $\mathcal{T}_a$,$\mathcal{T}_b$, the set of their timestamps $TS = \{timestamp(p) \mid p \in \mathcal{T}_a \vee p \in \mathcal{T}_b\}$, a look-ahead time Δt, and $t_1, t_2 \in TS$ we say that the interval $[t_1, t_2]$ contains a *repeated avoidance*, written $repeated\_avoid_\delta(\mathcal{T}_a, \mathcal{T}_b, [t_1, t_2], type)$, when for any $t \in [t_1, t_2]$ there exists $t' \in TS$ such that $t \in [t', t'+\Delta t]$ and $(t', \_) \in avoidances_\delta(\mathcal{T}_a, \mathcal{T}_b)\}$. Moreover, $type = sup\{type' \mid t' \in [t_1, t_2] \cap TS \ \wedge \ (t', type') \in avoidances_\delta(\mathcal{T}_a, \mathcal{T}_b)\}$ where

$$sup(X) = \begin{cases} weak & if\ \forall x \in X\ x = weak \\ mutual & if\ \exists x \in X\ x = mutual \\ individual & otherwise \end{cases}$$

Hence, the compression of the set of avoidances consists of a set of *maximal* intervals satisfying the *repeated_ avoid* predicate. The type of a repeated avoidance is the upper bound of the types of the avoidances occurring in the associated maximal interval.

Definition 4.11 (Compression of the set of avoidances): Given two trajectory tracks $\mathcal{T}_a$,$\mathcal{T}_b$, the set of their timestamps $TS = \{timestamp(p) \mid p \in \mathcal{T}_a \vee p \in \mathcal{T}_b\}$, a look-ahead time Δt, the compression of the set $avoidances_\delta(\mathcal{T}_a, \mathcal{T}_b)$ is defined as follows:

$$compressed\_avoid_\delta(\mathcal{T}_a, \mathcal{T}_b) = \{([t_1, t_2], type) \mid t_1, t_2 \in TS \wedge repeated\_avoid_\delta(\mathcal{T}_a, \mathcal{T}_b, [t_1, t_2], type) \wedge [t_1, t_2] \ \text{ maximal}\}$$

5. Algorithmic Framework

In this section, first we present an algorithm for solving the avoidance search problem (Section 5.1) and then we make some considerations on the choice of the right parameters to detect avoidances. As a consequence, we propose two different strategies for using our algorithm: the *simple detector* and the *fused detector*. The simple detector consists of a single execution of the algorithm with a fixed set of parameters whereas the fused

detector consists of multiple executions of the algorithm with various parameter sets and the different results are merged into a single result set (Section 5.2).

5.1. *An Algorithm for Avoidance Detection*

In order to solve the problems posed in Section 4.2 we propose Algorithm 1 which, given a finite set of trajectory tracks $\mathcal{D}$, a meet threshold δ and a look-ahead time Δt, returns a set of avoidance behaviors A, consisting of tuples specifying a pair of trajectories, a compressed interval in which the avoidance is detected between such trajectories and the relative type of avoidance.

The algorithm starts by considering every possible pair of trajectory tracks in $\mathcal{T}$ and, for each pair, it determines the first and last *useful* sample timestamps with respect to the look-ahead time Δt, (functions *getFirstUsefulSampleTime* and *getLastUsefulSampleTime*, lines 4-7). More precisely, if t_m^{start} and t_m^{end} denote, respectively, the timestamps of the very first and last samples of a trajectory track T_m, then the timestamps of the first and last *useful* samples with respect to Δt are determined as $getFirstUsefulSampleTime(T_m, \Delta t) = min\{t \mid p \in T_m \ \wedge \ p = (x, y, t) \ \wedge \ t \geq t_m^{start} + \Delta t\}$ and $getLastUsefulSampleTime(T_m, \Delta t) = max\{t \mid p \in T_m \ \wedge \ p = (x, y, t) \ \wedge \ t \leq t_m^{end} - \Delta t\}$.

Once the first and last useful samples of both trajectory tracks are determined (if any), the algorithm starts scanning the trajectories from these samples (line 8). The sample with the smaller timestamp, t^{curr}, is chosen (lines 9-12). Then, the algorithm proceeds with the avoidance search. First, it checks whether there is a *weak* avoidance in the interval $[t^{curr}, t^{curr} + \Delta t]$ (line 13): if there is an avoidance the algorithm determines whether the avoidance is mutual (line 15) or individual (line 17) by properly testing the *change-behavior* predicate. It could happen that there is not enough evidence to further specify the avoidance, hence it remains labeled as a *weak* avoidance.

Once the avoidance has been detected, the algorithm removes the last avoidance, identified by $[t_1, t_2]$ with type *typelast*, for the two trajectories under investigation T_m, T_n (line 19). It checks whether the interval $[t_1, t_2 + \Delta t]$ overlaps the interval $[t^{curr}, t^{curr} + \Delta t]$ and in this case it merges the two avoidances, inserting in the result set the interval $[t_1, t^{curr}]$ associated with the least upper bound between *typelast* and *type* (line 21). The definition of *sup* is given in Definition 4.10. Otherwise the algorithm inserts back the avoidance $[t_1, t_2]$ with type *typelast* in the result set and it adds also the new avoidance $[t^{curr}, t^{curr}]$ with *type* (line 23-24).

After having processed a sample, the function *nextSampleTime* returns the timestamp of the next sample(s) in the trajectory track(s) (lines 25-31).

As a final consideration we observe that our algorithm can be easily adapted to more complex processing pipelines, where preprocessing phases are allowed to filter out unnecessary information prior to the avoidance behavior detection phase, hence reducing the overall amount of data to be analyzed. Moreover, the algorithm can be also parallelized with respect to the set of trajectory pairs under investigation, since each pair can be processed separately in a core, thus entailing linear speedups with respect to the number of cores used.

5.2. *Avoidance Detectors*

In this section we present two different ways of using the algorithm proposed in the previous section for avoidance detection: *single detector* and *fused detector*. *Single detector* represents a single run of the algorithm using a fixed pair of $(\Delta t, \delta)$ values.

Algorithm 1: AVOIDANCE BEHAVIOR DETECTION

Input: Finite set of trajectory tracks $\mathcal{D}$, meet threshold δ, look-ahead time Δt.
Output: Set of avoidance behaviors A.

begin
  $A = \emptyset$
  foreach $T_m, T_n \in \mathcal{D}$ *with* $m < n$ **do**
    $t_m^{curr} = getFirstUsefulSampleTime(T_m, \Delta t)$
    $t_n^{curr} = getFirstUsefulSampleTime(T_n, \Delta t)$
    $t_m^{last} = getLastUsefulSampleTime(T_m, \Delta t)$
    $t_n^{last} = getLastUsefulSampleTime(T_n, \Delta t)$
    while $(t_m^{curr} < t_m^{last} \wedge t_n^{curr} < t_n^{last})$ **do**
      if $(t_m^{curr} < t_n^{curr})$ **then**
        $t^{curr} = t_m^{curr}$
      else
        $t^{curr} = t_n^{curr}$
      if $(avoid_\delta(interp(T_m), interp(T_n), [t^{curr}, t^{curr} + \Delta t]))$ **then**
        $type = weak$
        if $(change\_behavior(interp(T_m), [t^{curr}, t^{curr} + \Delta t]) \wedge$ $change\_behavior(interp(T_n), [t^{curr}, t^{curr} + \Delta t]))$ **then**
          $type = mutual$
        else if $change\_behavior(interp(T_m), [t^{curr}, t^{curr} + \Delta t]) \vee$ $change\_behavior(interp(T_n), [t^{curr}, t^{curr} + \Delta t])$ **then**
          $type = individual$
        $([t_1, t_2], typelast) = poplastResult(A, T_m, T_n)$
        if $t_2 + \Delta t >= t^{curr}$ **then**
          $addToResultSet(A, T_m, T_n, [t_1, t^{curr}], sup\{typelast, type\})$
        else
          $addToResultSet(A, T_m, T_n, [t_1, t_2], typelast)$
          $addToResultSet(A, T_m, T_n, [t^{curr}, t^{curr}], type)$
      if $t_m^{curr} < t_n^{curr}$ **then**
        $t_m^{curr} = nextSampleTime(T_m, t_m^{curr})$
      else if $t_n^{curr} < t_m^{curr}$ **then**
        $t_n^{curr} = nextSampleTime(T_n, t_n^{curr})$
      else
        $t_m^{curr} = nextSampleTime(T_m, t_m^{curr})$
        $t_n^{curr} = nextSampleTime(T_n, t_n^{curr})$
  return A

Given a pair of trajectory tracks ($\mathcal{T}_a$,$\mathcal{T}_b$), a meet threshold δ and a lookahead time Δ_t, the **single detector** returns the (possibly empty) set:

$$\mathcal{RS}^{\delta}_{\Delta_t} = \{I_i\}_{i=1,\ldots,m} = \{[t_s^i, t_e^i]\}_{i=1,\ldots,m} \tag{1}$$

where t_s^i and t_e^i (with $t_s^i < t_e^i$) denote the *starting* and *ending* timestamps of the i–*th*

avoidance, $I_i = [t_s^i, t_e^i]$ denotes a temporal interval during which an avoidance occurs, while the set $\{I_i\}_{i=1,\ldots,m}$ consists of *disjoint intervals*, i.e., $\forall i, j,\ I_i \cap I_j = \emptyset$.

Depending on the set of parameter values, an avoidance can be detected or missed. To minimize this problem, we propose a fusion detector that will detect the avoidance with different sets of parameters, and fuse the results in a unique output. We call this the **fused detector**.

Given a fixed lookahead time Δ_t, and a monotonically increasing sequence of meet thresholds $\langle \delta_1, \ldots, \delta_h \rangle$, where $\delta_1 < \ldots \delta_i < \ldots < \delta_h$, we can *fuse* the result sets obtained for each δ_i, still obtaining a disjoint set of temporal intervals. Specifically,

$$\mathcal{RS}_{\Delta_t}^{\langle \delta_1,\ldots,\delta_h \rangle} = \biguplus_{j=1,\ldots,h} \mathcal{RS}_{\Delta_t}^{\delta_j} \tag{2}$$

where the result set $\mathcal{RS}_{\Delta_t}^{\langle \delta_1,\ldots,\delta_h \rangle}$ is obtained by *fusing* the interval sets obtained by all the single detectors with parameters $\delta \in \{\delta_1, \ldots, \delta_h\}$. The operation $\biguplus$ is a simple set-union of the various intervals, except that the groups of *overlapping intervals* – such that, for each interval I in a group, there is at least another interval I' belonging to the same group, where $I \cap I' \neq \emptyset$ – are fused and replaced by a single larger interval, that spans all the overlapping ones. Note that this guarantees that the final fused result set is still composed of disjoint intervals, each one representing a distinct avoidance.

6. Experimental Evaluation

The goal of this section is to evaluate the proposed algorithm under different points of view. First, we quantitatively test the *effectiveness* of the algorithm by analyzing the ability to correctly detect expected avoidance behaviors. To this end we use an ad-hoc annotated dataset, our ground truth, created for the purposes of this work (Section 6.1). Second, we assess the ability of the algorithm in highlighting interesting and previously unknown patterns emerging from avoidance behaviors when using datasets for which no prior knowledge related to avoidance behaviors is available (Section 6.2).

6.1. *Experiments with the Ground-Truth Dataset*

We exploit a *ground truth* of annotated trajectories, explicitly created for this work, whose data derive from real GPS observations of moving objects collected in Florianopolis and Venice. The dataset contains an overall amount of 86 trajectories representing pedestrian movements, for a total of 7,834 samples and an average sampling rate of one second. Although this dataset is not large, it is sufficient to validate the proposed method.

In this trajectory dataset, 32 pairs of trajectories are labeled as *positive*, since they exhibit at least one avoidance. Among the positive pairs, 8 pairs exhibit two distinct avoidances while the remaining ones a single avoidance.

For each positive pair, the set of intervals during which the avoidance(s) occur(s) is reported as well. The positive/negative labels and the temporal intervals referring to single avoidances were given by human assessors. This may result in imprecise annotation of temporal intervals, since their span may depend on the perception of assessors.

In order to evaluate the effectiveness of the (simple/fused) detector that solves the *decision problem* (Definition 4.7), for each pair of trajectories we check the correctness of

the detector by verifying if the yes/no answer matches the positive/negative label in the ground truth. The detector returns *yes* if a non-empty set of avoidances is detected, *no* otherwise.

On the other hand, in order to evaluate the quality of the detector that solves the *search problem* (Definitions 4.9 and 4.11), for each positive pair correctly detected we also inspect the temporal intervals returned by the detector, by comparing them with the ones associated by the human assessor in the ground truth.

In the following we define the metrics used to evaluate the proposed method.

Decision problem. For this study we recur to well-established metrics commonly used in data mining to evaluate the quality of a classifier (Tan *et al.* 2005). Given a set of N pairs of trajectories, we evaluate the results of the detector algorithm by constructing an integer *confusion matrix* (see Table 1), a 2×2 table where the number of *true positives* (tp) and *true negatives* (tn) are given in the main diagonal, while the anti-diagonal contains the number of *false positives* (fp) and *false negatives* (fn) detected. Clearly, $N = tp + tn + fp + fn$.

We call *positive* any trajectory pair in the ground truth that is labeled as *avoidance behavior* = *yes*, while we use the term *negative* otherwise. Hence, tp and tn correspond to the pairs which the detector labels correctly *yes* or *no*, respectively, while fp and fn correspond to mislabeled pairs. Specifically, fp (fn) are pairs that the detector labels as positive (negative), but in fact appear as negative (positive) in the ground truth.

		Detected	
		positive	*negative*
Actual	*positive*	tp	fn
	negative	fp	tn

Table 1. Confusion matrix.

Recall and *Precision* are two widely used metrics employed in applications where successful detection of positive cases, i.e., in our case trajectory pairs for which an avoidance behavior is observed, is considered more significant than detection of other behavior. A formal definition of these metrics is given below:

$$\text{Precision,}\quad p = \frac{tp}{tp + fp} \qquad\qquad \text{Recall,}\quad r = \frac{tp}{tp + fn}$$

Precision p and recall r can be summarized into another metric known as *F-Measure*, defined as follows:

$$\text{F-Measure,}\quad F = \frac{2.r.p}{r + p}$$

where $0 \leq F \leq 1$.

Search problem. As previously stated, for positive pairs correctly detected by the simple/fused detector we also inspect the temporal intervals returned by comparing them with the ones associated by human assessor in the ground truth.

Let TP be the set of positive pairs p_i in the ground truth that were correctly identified by the detector, i.e., pairs p_i for which the result set returned by the algorithm is not

empty. Assuming that we are using a given combination Δt and δ, for clarity purposes in this context we denote such result set as $\mathcal{RS}^i$, omitting the parameters symbols in the notation. For each pair $p_i \in TP$, we know the set of actual disjoint temporal intervals $\mathcal{G}^i = \{\overline{I}^i_1, \ldots, \overline{I}^i_k\}$ in the ground truth, associated with $k > 0$ avoidances. The detector, either single or fused, also returns a set of m intervals $\mathcal{RS}^i = \{I^i_1, \ldots, I^i_m\}$.

Let $\widehat{\mathcal{G}}^i = \{\overline{I}^i_j \in \mathcal{G}^i \mid \exists! I^i_h \in \mathcal{RS}^i \; s.t. \; (\overline{I}^i_j \cap I^i_h \neq \emptyset) \; \wedge \; (\nexists \overline{I}^i_k \in \mathcal{G}^i, k \neq j \; s.t. \; \overline{I}^i_k \cap I^i_h \neq \emptyset)\}$, $\widehat{\mathcal{G}}^i \subseteq \mathcal{G}^i$, be the set of intervals in $\mathcal{G}^i$ such that each interval overlaps with *only one* interval in $\mathcal{RS}^i$, and the latter does not overlap with any other intervals in $\mathcal{G}^i$.

Finally, we can quantitatively evaluate the quality of the results for all the pairs $p_i \in TP$ by *Q-Measure*:

$$Q\text{-}Measure = \frac{\sum_{p_i \in tp} \frac{|\widehat{\mathcal{G}}^i|}{|\mathcal{G}^i|}}{|tp|}$$

where $0 \leq Q\text{-}Measure \leq 1$. Ideally *Q-Measure* should be equal or close to 1.

In the following we present some visual examples of the output of the algorithm (Section 6.1.1), then in Sections 6.1.2 and 6.1.3 we show quantitatively the results obtained by exploiting the (simple/fused) detector that solves the decision or the search problem.

In all the experiments described below, we discuss the various results obtained by changing the two main parameters of our avoidance detection algorithm, namely Δt and δ, whose values are reported in seconds and meters, respectively.

6.1.1. Visual inspection of avoidances

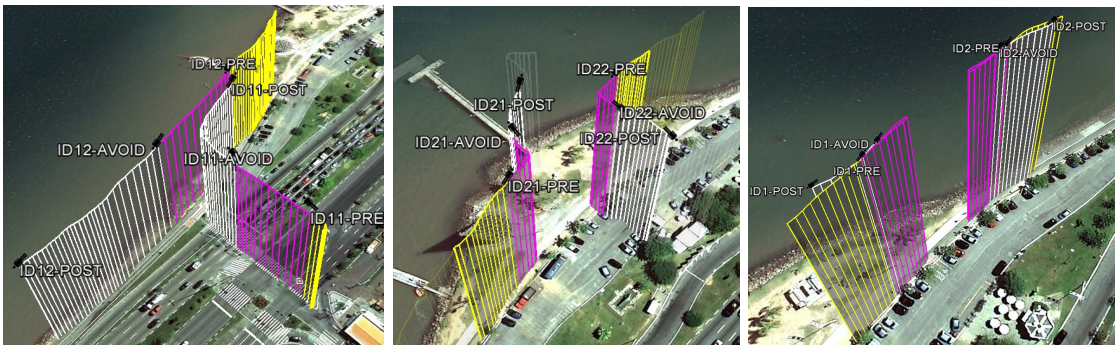
In order to visualize some avoidances detected by our algorithm on the ground truth dataset, we conveniently use Google Earth. Specifically, the avoidances shown in Figure 8 refer to trajectory pairs collected in Florianopolis. The subset of segments highlighted in *purple* represents the set of samples over which an avoidance is detected. The segments highlighted in *yellow* and *white* represent, respectively, a fixed sequence of samples occurring *before* and *after* the avoidance detected by the algorithm. Figure 8(a) represents an individual avoidance by entity ID11 (which moves initially from bottom-right to top-left), slowing down at some point in order to avoid ID12 (moving from top-right to bottom-left). Figure 8(b) depicts a mutual avoidance where ID21 (moving from bottom-left to top-right) and ID22 (moving in the opposite direction) change their direction as soon as they get close. Finally, Figure 8(c) depicts a mutual avoidance where the two entities invert their direction as soon as they get too close.

6.1.2. Results for the Decision Problem

In this section we evaluate the ability of the detectors (simple detector and fused detector) of correctly identifying the trajectory pairs of the ground truth that are positively/negatively labeled (decision problem).

Simple detector. For each pair of parameters Δt and δ, we build the confusion matrix by considering all trajectories in the ground truth, then determine precision/recall, and finally compute the F-Measure scores. Figure 9 reports the scores obtained by the algorithm for all combinations of parameters. Specifically, each curve in the plot refers to a distinct Δt, and shows the F-Measure score as a function of δ.

For almost all values of Δt, a common optimal value for the meet threshold δ that maximizes the F-measure score falls in the interval $[3, 6]$; for larger values of δ, the score degrades. It is worth noting that these optimal values for δ approximately reflect the aver-



(a) (b) (c)

Figure 8. Examples of three visual inspections performed on three different avoidances returned by the algorithm.

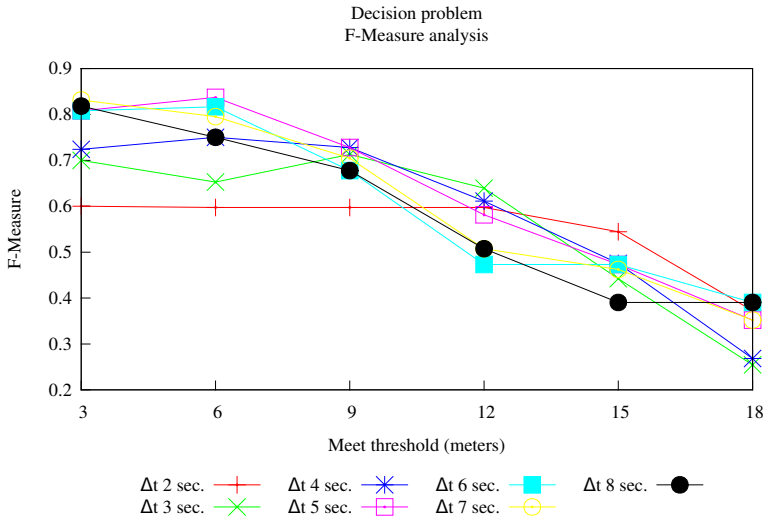


Figure 9. Decision problem with simple detector: F-Measure analysis.

age avoidance distance used to physically produce the avoidances for (positive) trajectory pairs included in the ground truth.

Fused detector. In the following we show how the overall performance of the algorithm can substantially be improved by conveniently *fusing* different result sets of distinct simple detectors, according to the fusion operator defined by Equation (2).

Given a Δt, and a sequence of values for δ, e.g., $\langle 3, 6, 9\rangle$, a pair of trajectories is identified as a positive case, i.e., *avoidance behavior* = *yes*, if at least a simple detector for some δ in the sequence identifies one or more avoidance behaviors. Conversely, if no simple detector is able to recognize any avoidance, the pair is identified as a negative case, i.e., *avoidance behavior* = *no*. Still, for the fused detector we can build the confusion matrix, by considering all trajectory pairs in the ground truth, and finally compute the F-Measure scores.

Considering the results shown in Figure 9, we choose $\Delta t \in \{4, 5, 6, 7\}$ to evaluate the fused detector. For each Δt, we compute the F-Measure related to the fused result sets $\mathcal{RS}_{\Delta t}^{\langle\delta=3\rangle}, \mathcal{RS}_{\Delta t}^{\langle\delta=3,\delta=6\rangle}, ..., \mathcal{RS}_{\Delta t}^{\langle\delta=3,\delta=6,...,\delta=18\rangle}$ (each one defined as per Equation 2).

Results are reported in Figure 10, where each fused result set is represented by its upper δ threshold in the X-axis. The figure shows how the fused detector entails substantial improvements in terms of F-Measure (up to 95%), provided that Δt is properly chosen according to the dataset features.

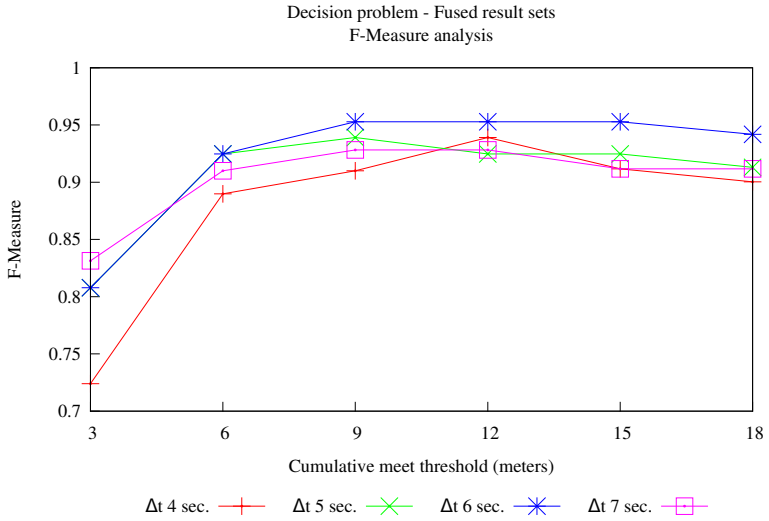


Figure 10. Decision problem with fused detector: F-Measure analysis. In X-axis the values are the maximums of the thresholds δ used during the fusion operation, e.g., 9 is the maximum for the set $\{3, 6, 9\}$.

In general, we argue that the opportunity of fusing different result sets depends on the kind of analysis we want to perform. Specifically, it depends on the classes of avoidance behaviors we want to discover (e.g., only values for δ that are relevant for our purposes should be used for the fusion operation), and on the amount of useful information an analyst is interested in extracting at the expense of possible losses in precision (due to the detection of false avoidances).

6.1.3. Results for the Search Problem

In this section we assess the quality of the temporal intervals reported by both detectors (simple and fused), for the positive pairs correctly detected, and thus included in the set of trajectory pairs TP, where $tp = |TP|$.

Simple detector. Figure 11 reports the Q-Measure scores obtained in the experiments. The experimental findings confirm all the remarks done in Section 6.1.2 concerning Δt and δ. In general, we obtain the best Q-Measure score for the same parameters Δt and δ for which we obtained the best F-Measure scores. We also point out that using high values of Δt (where *high* is relative to the dataset features) may erroneously induce the fusion of distinct avoidance behaviors, due to compression, thus potentially mapping multiple avoidances occurring between two trajectories in the ground truth to a single detected avoidance. This in turn induces losses in terms of Q-Measure scores.

Fused detector. Also for the fused detector we aim at analyzing the quality of the temporal intervals for the trajectory pairs in TP. To this end, we consider again the fused result sets belonging to $\{\mathcal{RS}_{\Delta t}^{\langle\delta=3\rangle}, \mathcal{RS}_{\Delta t}^{\langle\delta=3,\delta=6\rangle}, ..., \mathcal{RS}_{\Delta t}^{\langle\delta=3,\delta=6,...,\delta=18\rangle}\}$.

Figure 12 reports the performance of the algorithm in terms of Q-Measure score. We can observe how fusing different result sets entails substantial improvements. These im-

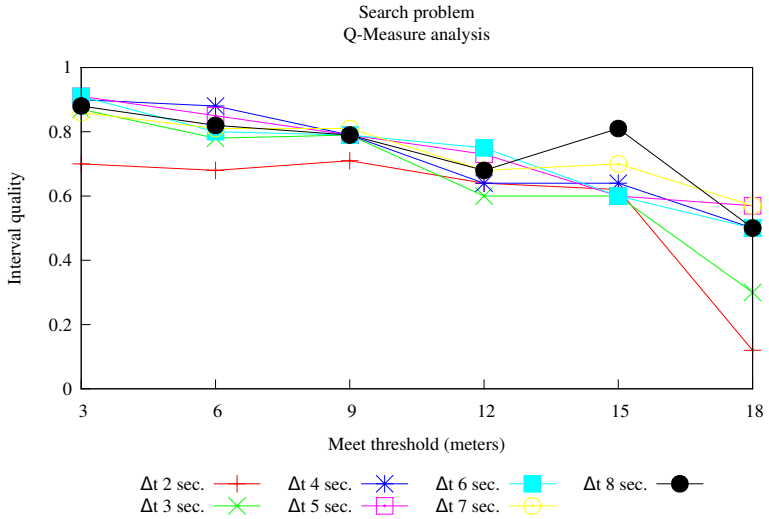


Figure 11. Search problem with simple detector: Q-Measure analysis.

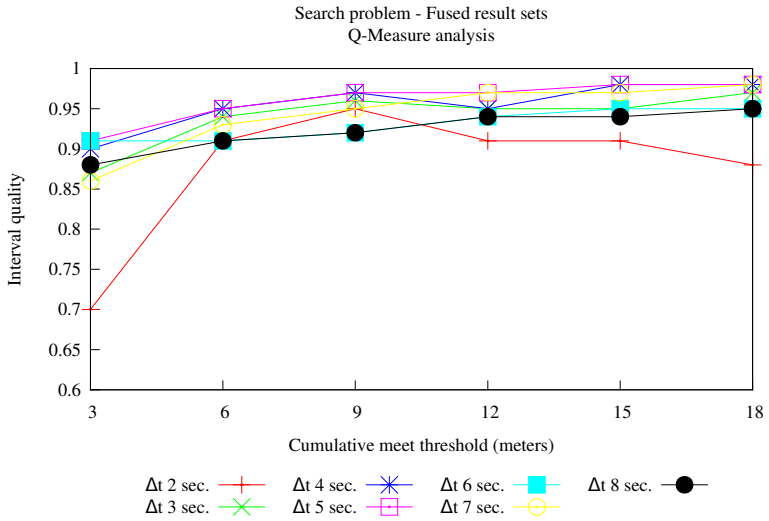


Figure 12. Search problem with fused detector: Q-Measure analysis. In X-axis the values are the maximums of the thresholds δ used during the fusion operation, e.g., 9 is the maximum for the set $\{3, 6, 9\}$.

provements are particularly evident whenever the fusion is performed for δ values close to the average distances used for physically producing avoidances ($\delta \in [3, 6]$).

6.2. *Analysis of a Real World Unannotated Dataset*

In this section we consider the AIS Brest dataset, a real world unannotated dataset containing 824 trajectories related to the movements of 824 ships[1] nearby Brest's harbor (Etienne *et al.* 2012). Basic statistics reveal that the dataset contains 5.756.438 points, the

[1]Each trajectory is uniquely associated with a ship.

trajectories move at an average speed of 7.77 km/h and most of the trajectories have an average sampling rate between 1 and 20 seconds. These characteristics make the dataset quite interesting in terms of the precision with which the trajectories are described.

By considering the aforementioned statistics, and after a scrutinization of different meet thresholds and look-ahead times, according to entities' features we chose a meet threshold δ of 30 meters and a look-ahead time Δt of 50 seconds. Indeed, we argue that for this case study this combination of parameters allows us to capture interesting patterns, as we will show further on.

The algorithm detects a total of 1480 avoidances, among which 321 are mutual, 970 individual and 189 weak. Given this considerable amount of information it is necessary to perform a deeper analysis in order to infer meaningful patterns.

Among the 824 ships, 229 are involved in at least one avoidance. We call this set as the *set of active ships*. If we further look at the number of avoidances in which each active ship is involved, we notice that 8 ships are involved in more than 100 avoidances, while the vast majority - more precisely 196 ships (which constitutes the 85,5% of the active ships set) - are involved in a number of avoidances between 1 and 10. We call the former set as the *set of frequent ships* while the rest of the ships ends up in the set of *infrequent ships*.

The information above suggests that the frequent ships play a very important role in the dataset. If we decompose the total amount of avoidances detected by the algorithm, we find out that the overall amount of avoidances between frequent and infrequent ships are 973 (65,7% of the result set), while the avoidances between frequent ships are 386 (26,1%) and between infrequent ones are 121 (8,2%).

If we look at the MMSI codes of the frequent ships in order to find out their typology (Table 2), we have that the top-2 frequent ships are *pilot ships*, while the remaining ones are *passenger ships* and *tugboats*.

MMSI Code	*Type*	Amount of avoidances
227730220	Pilot ship	414
227005550	Pilot ship	364
227635210	Passenger ship	194
227592820	Passenger ship	175
227574020	Passenger ship	174
227612860	Passenger ship	158
227574030	Passenger ship	147
228051000	Tugboat	119

Table 2. *Frequent ships* details.

Given these data we want to answer the following questions:

(1) Which are the events producing so many avoidances between frequent and infrequent ships?
(2) Which are the events producing a considerable amount of avoidances between frequent ships?
(3) Is there any kind of recurring pattern causing avoidances between infrequent ships?

When answering Question (1) we notice a dominant pattern (Figure 13) that we call *paired movement event*. Through a graphical inspection we observe that almost all these events can be decomposed in 3 phases: during the first phase the two ships approach each other, mostly when they are entering or exiting the harbor (*approach* phase, Figure 13(a)).

(a) Approach (b) Paired movement (c) Detach

Figure 13. Example of a paired movement event involving the frequent ship 228051000. The ships are moving from bottom to top.

Then, the ships proceed paired (*paired movement* phase, Figure 13(b)) until they approach the docks or they exit the harbor area. During this intermediate phase some avoidances may emerge or not, depending on the continuous adjustment performed by both ships in order to maintain the relative distance. Finally, the ships separate (*detach* phase), as shown in Figure 13(c).

Given the typologies of the frequent ships reported in Table 2 we argue that the pilot ships and the tugboats produce these events when they have to pilot (or tow, respectively) an incoming (or outgoing) ship. For what concerns the passenger ships, we argue that they adjust their trajectories in order to avoid other infrequent ships nearby; moreover, the amount of avoidances in which they are involved is justified by the fact that they are servicing, and thus repeatedly going through, a fixed route.

In general we expect that these avoidances are mostly distributed in predetermined areas. Indeed, if we plot the avoidances occurring during a two-month window we see that they are approximately distributed on a fixed path going from the harbor's docks to the strait exit (Figure 14). This also gives an idea about the most dangerous or trafficked areas (especially near the docks).

Figure 14. Subtrajectories related to avoidance behaviors detected in a time interval spanning two months ([20/04/2009, 20/06/2009]).

Concerning Question (2), we found that many avoidances are produced according to the same pattern observed for Question (1), or when a frequent ship is docking (and therefore slowing down, hence the avoidance) in the harbor nearby already docked ships.

The latter pattern is observed between frequent and infrequent ships as well, although with a lesser extent.

Finally, as far as Question (3) is concerned we found out that the second pattern observed when explaining the avoidances related to Question (2) also occurs, i.e., almost all the avoidances between infrequent ships happen near the docks when one or more ships are docked while one ship is docking nearby.

7. Conclusion

Several algorithms have been proposed for mining different types of trajectory patterns. However, an interesting behavior that has not been much explored in historical trajectories of moving objects is *avoidance*.

In this paper we have introduced a new type of trajectory pattern: avoidance between moving objects. We presented a set of theoretical definitions and an algorithm which is able to detect such a pattern. The discovery of avoidances between moving objects is challenging, since the intent of any moving object may not be immediately apparent from its trajectories. To determine an avoidance, two objects should move towards the same area at the same time, but either one or both should change their behavior when they come close enough to be aware of each other. To identify a behavior change we forecast the movements of both moving objects and compare them with the actual movements. If the forecasts predict a meet but the actual movements do not meet, an avoidance is detected. Each detected avoidance is in turn classified, whenever possible, as individual when only one moving object changes significantly its behavior, or as mutual when both objects change their behavior significantly.

It is worth mentioning that a behavior change is measured through the distance between the forecast movement and the actual movement. Such generalization prevents from using specific features such as direction and speed for detecting a change of behavior.

In this paper, besides analyzing the parameters of the algorithm, we went one step further by introducing the idea of fused detector, which merges the result sets of several simple detectors (with different meet thresholds) in order to allow the detection of a possibly broad range of avoidance behaviors. The proposed approach for avoidance detection makes use of only two parameters and is able to deal with trajectories collected at different sampling rates and/or having different temporal lengths.

The algorithm has been evaluated with two real-world datasets. The first dataset is annotated and it contains pedestrian movements; the purpose of analyzing such dataset is to verify that the algorithm was able to correctly detect avoidances which actually occur. The method correctly detected the avoidances (F-measure up to 95%). The second real-world dataset, unannotated, contains ship movements nearby the Brest's harbor. Since no prior avoidance information is available, the purpose of analyzing such dataset is to check whether the algorithm is able to extract interesting evidence from the data. Indeed, by characterizing the avoidances between ships on the basis of their frequency, their spatial distribution and by means of visual inspections on the behavior of frequent ships, we were able to highlight the most trafficked areas, as well as a frequently recurring event, i.e., the paired movement event.

We have not compared the results of our algorithm with other approaches because, to the best of our knowledge, there is no other work for avoidance detection that identifies every avoidance occurrence in a trajectory dataset.

Future work includes an analysis on the effect of using different forecast functions and

the definition of a confidence measure to evaluate the avoidance.

References

Alvares, L.O., *et al.*, 2011. An algorithm to identify avoidance behavior in moving object trajectories. *J. Braz. Comp. Soc.*, 17 (3), 193–203.

Borenstein, J. and Koren, Y., 1991. The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, 7 (3), 278–288.

Brooks, R.A., 1986. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2 (1), 14–23.

de Lucca Siqueira, F. and Bogorny, V., 2011. Discovering Chasing Behavior in Moving Object Trajectories. *Transactions in GIS*, 15 (5), 667–688.

Dodge, S., Weibel, R., and Lautenschütz, A.K., 2008. Towards a taxonomy of movement patterns. *Information Visualization*, 7 (3-4), 240–252.

Etienne, L., Devogele, T., and Bouju, A., 2012. Spatio-temporal trajectory analysis of mobile objects following the same itinerary. *Advances in Geo-Spatial Information Science*, 10, 47.

Giannotti, F., *et al.*, 2007. Trajectory pattern mining. *In*: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, 330–339.

Gudmundsson, J. and van Kreveld, M.J., 2006. Computing longest duration flocks in trajectory data. *In*: R.A. de By and S. Nittel, eds. *14th ACM International Symposium on Geographic Information Systems, ACM-GIS 2006, November 10-11, 2006, Arlington, Virginia, USA, Proceedings* ACM, 35–42.

Hafner, M.R., *et al.*, 2013. Cooperative Collision Avoidance at Intersections: Algorithms and Experiments. *IEEE Transactions on Intelligent Transportation Systems*, 14 (3), 1162–1175.

Khansari-Zadeh, S.M. and Billard, A., 2012. A Dynamical System Approach to Realtime Obstacle Avoidance. *Autonomous Robots*, 32 (4), 433–454.

Khatib, O., 1986. Real-time obstacle avoidance for robot manipulator and mobile robots. *International Journal of Robotics Research*, 5 (1), 90–98.

Kim, D.J., Park, K.H., and Bien, Z., 2007. Hierarchical longitudinal controller for rear-end collision avoidance. *IEEE Transactions on Industrial Electronics*, 54 (2), 805–817.

Laube, P., Imfeld, S., and Weibel, R., 2005. Discovering relative motion patterns in groups of moving point objects. *International Journal of Geographical Information Science*, 19 (6), 639–668.

Li, Z., *et al.*, 2010. Mining periodic behaviors for moving objects. *In*: B. Rao, B. Krishnapuram, A. Tomkins and Q. Yang, eds. *KDD* ACM, 1099–1108.

Li, Z., *et al.*, 2013. Attraction and Avoidance Detection from Movements. *PVLDB*, 7 (3), 157–168.

Liu, Y.H. and Shi, C.J., 2005. A fuzzy-neural inference network for ship collision avoidance. *In*: *Proceedings of 2005 International Conference on Machine Learning and Cybernetics*, Guangzhou, China IEEE Computer Society, 4754–4754.

Mou, J.M., van der Tak, C., and Ligteringen, H., 2010. Study on collision avoidance in busy waterways by using AIS data. *Ocean Engineering*, 37 (5), 483–490.

Nedevschi, S., Bota, S., and Tomiuc, C., 2009. Stereo-based pedestrian detection for collision-avoidance applications. *Transactions on Intelligent Transportation Systems*, 10 (3), 380–391.

Pang, L.X., *et al.*, 2013. On detection of emerging anomalous traffic patterns using GPS

data. *Data Knowl. Eng.*, 87, 357–373.

Parent, C., *et al.*, 2013. Semantic Trajectories Modeling and Analysis. *ACM Computing Surveys*, 40.

Pechoucek, M. and Sislak, D., 2009. Agent-Based Approach to Free-Flight Planning, Control, and Simulation. *IEEE Intelligent Systems*, 24 (1), 14–17.

Renso, C., Spaccapietra, S., and Zimanyi, E., eds. , 2013. *Mobility Data: Modeling, Management, and Understanding*. Cambridge, UK: Cambridge University Press.

Richards, A. and How, J.P., 2002. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. *In*: *American Control Conference, 2002. Proceedings of the 2002*, Vol. 3, 1936–1941.

Shandy, S. and Valasek, J., 2001. Intelligent Agent for Aircraft Collision Avoidance. *In*: *Proceedings of AIAA Guidance, Navigation, and Control Conference*, Montreal, Canada American Institute of Aeronautics and Astronautics, 1–11.

Sun, D., Kleiner, A., and Nebel, B., 2014. Behavior-based Multi-Robot Collision Avoidance. *In*: *Proceedings of the IEEE International Conference on Robotics and Automation*, 1668–1673.

Tan, P.N., Steinbach, M., and Kumar, V., 2005. *Introduction to Data Mining, (First Edition)*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Trasarti, R., *et al.*, 2011. Mining mobility user profiles for car pooling. *In*: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1190–1198.

Wachowicz, M., *et al.*, 2011. Finding moving flock patterns among pedestrians through collective coherence. *International Journal of Geographical Information Science*, 25 (11), 1849–1864.

Xausa, I., *et al.*, 2012. Avoidance Trajectories for Driver Assistance Systems via Solvers for Optimal Control Problems. *In*: *International Symposium on Mathematical Theory of Networks and Systems* Springer, 1–8.