commsignia

Commsignia Safety Applications

Version: 23-01
Publication date January 26, 2023
Document ID: SAF-001

Confidentiality notice: This document is provided in confidence and may not be used for any purpose other than that for which it is supplied. All content within may not be disclosed to any third party or used for other purpose without the written permission of Commsignia Ltd.

This document may be revised without prior notice. For most recent releases, please visit our website.

Table of Contents

List of Figures

1. Introduction

The safety applications detailed in this document are Commsignia's solutions for covering the most common real-life use cases of V2X technology that enhance the traveling experience of all road users by making private and public transportation safer, more efficient, and economically sustainable.

Safety applications can be categorized by the underlying technology that is used to connect entities in traffic. “Vehicle-to-Vehicle (V2V) applications” (p. 4) rely on communication between vehicles (such as cars, emergency vehicles, public transportation, and even trains), whereas “Infrastructure-to-Vehicle (I2V) applications” (p. 27) enables vehicular communication with V2X equipped parts of the road infrastructure (such as Traffic Light Controllers in intersections or programmable roadside information signs along a highway).

Each application within these categories has its own chapter, describing the real-life use case it is related to and detailing the technical aspects of their filtering and triggering conditions where applicable. In certain cases Commsignia provides technical solutions that cover more than one use cases of V2X communication. In this document, each of these use case variants are detailed in the chapter of the related Safety Application.

2. Common notification contents

The data fields detailed in this chapter are commonly used by application notifications.

2.1. Object type

The object type is an enumerated type with the following elements:

- Unknown
- Vehicle
- Pedestrian
- Cyclist
- Animal
- Other

2.2. Vehicle type

Object types of Vehicle enumerators are further divided into the following elements:

- Unknown
- PassengerCar
- Special
- LightTruck
- Truck
- Motorcycle
- Transit
- FixedGuideway
- Emergency
- Infrastructure
- Other

2.3. Object data

Object Data represents an object in the global coordinate space. The data consists of the following fields:

- The position of the object, in latitude/longitude format, with a confidence ellipse.
- The speed of the object, with a confidence interval.
- The heading of the object, with a confidence interval.
- The type of the object (see Object Type).
- The vehicle type of the object, if it is a vehicle (see Vehicle Type).

2.4. Relative data

The Relative Data container represents the relative position and motion of an object, compared to another object (usually the host vehicle). The data consists of the following fields:

- The relative position of the object, in X/Y format, with a confidence interval. X is the left–right and Y is the forward–backward axis.
- The distance of the object.
- The relative speed of the object, with a confidence interval.
- The relative heading of the object, with a confidence interval.

3. Speed-dependent behavior

Specific applications can be configured to be automatically enabled or disabled based on the speed of the host vehicle.

3.1. Speed-dependent configuration

These applications have the following configuration values under their own configuration key:

- vMin: The minimum speed where the application is always enabled;
- vMax: The maximum speed where the application is always enabled;
- vMinHyst: The hysteresis of the minimum speed;
- vMaxHyst: The hysteresis of the maximum speed.

3.2. Logic of speed-dependent behavior

Based on the above values, the applications are automatically enabled/disabled, as shown in Figure 1, as follows:

- If the speed of the host vehicle is unknown, the application is disabled.
- If the application is currently enabled and the speed of the host vehicle becomes less than vMin, it is disabled.
- If the application is currently enabled and the speed of the host vehicle becomes greater than vMax + vMaxHyst, it is disabled.
- If the application is currently disabled and the speed of the host vehicle becomes greater than vMin + vMinHyst, it is enabled.
- If the application is currently disabled and the speed of the host vehicle becomes less than vMax, it is enabled.

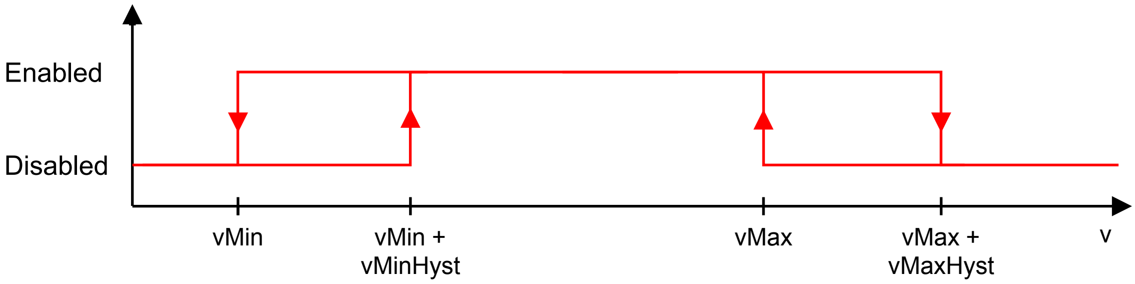

Figure 1. Logic of speed-dependent behavior

4. Vehicle-to-Vehicle (V2V) applications

4.1. Blind Spot Warning (BSW)

This safety application issues a warning about a possible collision.

4.1.1. BSW safety application overview

The Blind Spot Warning (BSW) safety application is a V2V communication based safety feature that issues a warning to the driver of the host vehicle in case of a possible collision with a remote vehicle in the same direction of travel in another lane and in the blind spot of the driver.

When traveling on a multi-lane road, the BSW application sends a notification to the driver of the host vehicle when a remote car is positioned in their blind spot. This application is compliant with and relies on the European (ETSI G5) and the US (IEEE WAVE) standards.

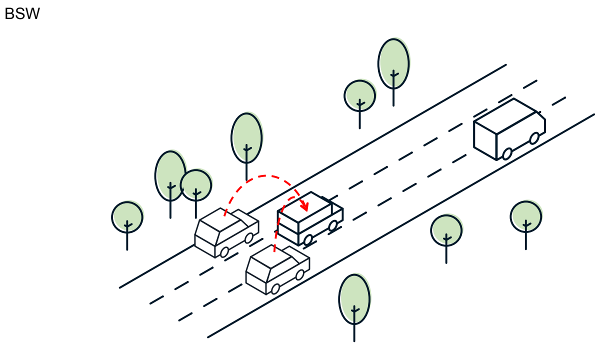

Figure 2. BSW Safety Application

4.1.2. BSW mode of operation

Model

- Blind Spot Warning (BSW) is a V2V safety application that is enabled or disabled based on the speed of the host vehicle (for more information, see “Speed-dependent behavior” (p. 3)).
- The application uses the [Object] entity type.
- The application uses the [Lane abstraction] classifiers to identify a remote vehicle in the target areas on both sides of the host vehicle and triggers based on the time spent by the remote vehicle in one of the target areas.

Filter setup

The application identifies vehicles with identical heading that are within the defined target areas behind the vehicle on both sides according to the following criteria:

- The remote vehicle is in [one lane left]or [one lane right.]
- The remote vehicle is at least **targetOffset** distance behind (in [m]) or ahead if **targetOffset** is negative.
- The remote vehicle is at most **(targetOffset + targetLength)** distance behind.
- The remote vehicle is traveling in the [same direction.]
- The remote vehicle is at the [same altitude.]

Triggering conditions

The application sends a warning if one or more remote vehicle spends more than **timeout** time in the target area. The warning provides information about the entering time of the target vehicle into the defined dangerous zone and the duration in which the vehicle will be inside this area.

4.1.3. BSW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.bsw: true**
- The following parameters can be configured in the **bsw** key:
  - **targetLength**: Length of the target area in [m]
  - **targetOffset**: Offset of target front from the reference point of the host vehicle in [m]
  - **timeout**: Waiting time before the warning is triggered by the application [s]

Common configuration parameters affecting the application in **cff.json**:

- [Lane abstraction] configuration in **core.classification.lane.width** member
- [Same direction] classifier configuration in **core.classification.headingDiffThr** member
- [Same altitude] classifier configuration in **core.classification.altitudeDiffThr** member

4.1.4. BSW notifications

The application sends ALERT level notifications, which contain the Object Data of the host vehicle and the Object Data and Relative Data of the remote vehicle (for more information, see “Common notification contents” (p. 2)).

4.2. Control Loss Warning (CLW)

This safety application issues a warning about a possible collision hazard with a vehicle that is out of control.

4.2.1. CLW safety application overview

The Control Loss Warning safety application is a V2V communication based safety feature that issues a warning to the driver of the host vehicle in case of a possible collision with a remote vehicle that is broadcasting information about loss of control.

The application triggers whether the remote vehicle is traveling in the same or in the opposite direction as the host vehicle. The CLW helps drivers to avoid or mitigate front-to-rear or front-to-front vehicle collision in the forward path of travel. The application sends a notification alert that contains vehicle data as well as Time-To-Collison (TTC).

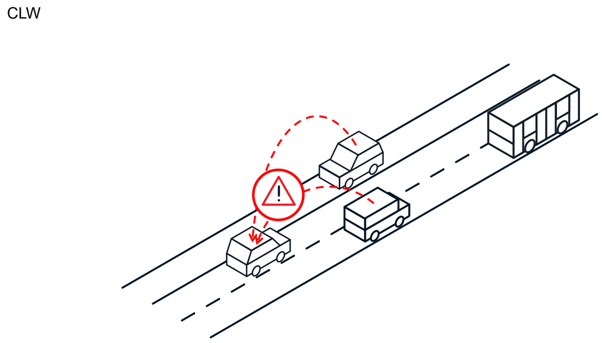

Figure 3. CLW Safety Application

4.2.2. CLW mode of operation

Model	• CLW is a V2V safety application that is enabled or disabled based on the speend of the host vehicle (for more information, see “Speed-dependent behavior” (p. 3)). • If enabled, the **targetLength** value can be changed based on the speed of the host vehicle. The effective **targetLength** is equal to **tar-**

getLength × (host speed [km/h]/50 [km/h]). This value must be in the range of 10–1000 m.

- If enabled, the maximum TTC value can be changed based on the speed of the host vehicle. The effective maximum TTC is equal to maximum TTC × (host speed [km/h]/50 [km/h]). If this computed effective value is less than 3 s, then it is set to be equal to 3 s.
- The application uses the [Object] entity type.
- The application uses the [Lane abstraction] classifiers to identify a remote vehicle in the target areas on both sides of the host vehicle and triggers based on the time spent by the remote vehicle in one of the target areas.

Filter setup

The application subscribes to remote [Objects] that match the following criteria:

- The remote vehicle is [within one lane] (either in the [same lane] or [one lane left]or [one lane right).]
- The remote vehicle is at most **targetLength** distance ahead.

> **NOTE**
>
> Using target length to "pre-filter" objects is an optional optimization. The application works the same if the **targetLength** is a large value, because it calculates the actual predicted collision times. Keeping this value small or speed dependent can improve performance, because the application does not have to perform as many calculations. However, when setting this value, it needs to be set larger, rather than smaller to avoid false negatives.

.

- The remote vehicle is broadcasting ABS, SCS, or Traction Control activation.
- The remote vehicle is at the [same altitude.]

Triggering conditions

On every processing cycle, for every filtered object:

- The application computes the closest it will get to the object, assuming both are moving along a straight line, at either constant speed or constant acceleration depending on the configuration.
- The calculated closest distance is corrected using the length of both objects.
- If this distance is less than the configured minimum clearance distance and the time it takes to reach this distance is below the configured maximum TTC, a notification is sent.

4.2.3. CLW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.clw: true**
- The following parameters can be configured in the **clw** key:
  - **targetLength**: Length of the target area in [m].
  - **useAdaptiveTarget**: Use speed-dependent target length scaling.
  - **dangerousDistance**: The minimum clearance distance that is considered safe TTC: The maximum time it can take to reach the dangerous distance.
  - **useAdaptiveTtc**: Use speed-dependent TTC scaling. It is recommended to set either this or **useAccelerationForTtc** to true.

- **useAccelerationForTtc**: Assume constant acceleration when calculating the minimum distance point. It is recommended to set either this or **useAdaptiveTtc** to true.

Common configuration parameters affecting the application in **cff.json**:

- [Lane abstraction] configuration in **core.classification.lane.width** member
- [Same altitude] classifier configuration in **core.classification.altitudeDiffThr** member

4.2.4. CLW notifications

The application sends ALERT level notifications, which contain the Object Data of the host vehicle and the Object Data and Relative Data of the remote vehicle (for more information, see “Common notification contents” (p. 2)).

The application also sends Time-to-Collision information in the notification as well. This is the predicted time (in [ms]) until the two vehicles are closest to each other.

4.3. Do Not Pass Warning (DNP)

This safety application issues a warning about an impending frontal collision with another vehicle during an overtaking maneuver.

4.3.1. DNP safety application overview

The Do Not Pass Warning (DNP) application is a V2V communication based safety feature that issues a warning to the driver of the host vehicle in case of an impending frontal collision with a remote vehicle in the adjacent lane while performing an overtaking maneuver in the upstream traffic. The DNP helps drivers to avoid or mitigate front-to-front vehicle collision in the forward path of travel. This application is compliant and relies on to the European (ETSI G5) and the US (Wave) standards.

NOTE

This application only takes the right-side driving traffic use case into consideration.

DNP

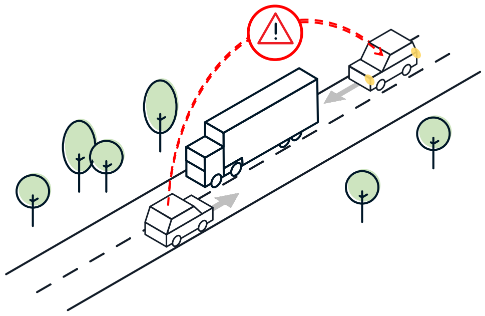

Figure 4. DNP Safety Application - the host vehicle is traveling behind the truck

4.3.2. DNP mode of operation

Model

- DNP is a V2V application that works for the **Object** remote entity type.
- The application is enabled or disabled based on the speed of the host vehicle (for more information, see “Speed-dependent behavior” (p. 3)).
- The application uses [lane abstraction] for target areas for ahead and oncoming objects.

- The application triggers based on complex analysis of the ahead and oncoming objects.
- Preconditions:
  - There is at least one ahead object. Note: a passing situation cannot exist without an object ahead.
  - The host vehicle must be [fast enough] compared to the objects ahead and the nearest object ahead is not faster than the host vehicle by more than a predefined constant.
  - The host is [close enough] to the objects ahead. The distance between the host and the nearest object ahead is shorter than the **activationDistance** or the following time of the host vehicle is less than **activationTimeGap**.

Filter setup

Filter setup for objects ahead:

- The remote vehicle is in the [same lane].
- The remote vehicle is going the [same direction].
- The remote vehicle is located [ahead].

Filter setup for oncoming objects:

- The remote vehicle is in [one lane left].
- The remote vehicle is going the [opposite direction].
- The remote vehicle is located [ahead].

Triggering conditions

The following calculations are handled on every processing cycle:

- The following preconditions are checked. If any of them fail, the processing stops for the cycle:
  - The closest ahead object must not be more than 2 m/s faster than the host vehicle.
  - The gap between the closest ahead object and host vehicle must not be more than activationDistance.
  - The following time (the gap divided by the speed of the host vehicle) between the host vehicle and the closest ahead object must not be more than activationTimeGap.
  - The host vehicle must be signaling to the left or the sendPreventiveInfo configuration option must be set.
- All ahead objects are accumulated and sorted from the closest to farthest. Only the closest 32 objects are kept.
- All oncoming objects are accumulated and sorted from the closest to farthest. Only the closest 32 objects are kept. Oncoming objects that are closer than the closest ahead object are dropped after accumulation.
- A merging point is calculated by finding a gap between ahead objects sufficiently big to merge into.
  - The gap length must be larger than 10 m + the length of the host vehicle.
  - The following time (the gap divided by the speed of the closer object) must be larger than the configured **mergingTimeGap**.
- If no suitable gap is found, it is considered to be after the last ahead object.
- The parameters of the passing maneuver are calculated from:
  - The relative distance to be passed (the full length of the to-be-passed line of objects) is calculated by adding together
    - the distance between the closest object and the last to-be-passed object
    - half the length of both objects
    - the length of the host vehicle

    - 10 m of clearance.
  - The net time it takes to pass the objects is calculated by dividing the length to be passed with the speed difference. If the speed difference between the last to-be-passed object and the host vehicle is less than **additionalOvertakingSpeed**, it is considered to be **additionalOvertakingSpeed** for the whole maneuver.
  - The full distance traveled by the host vehicle during the maneuver is calculated by adding together
    - the relative distance to be passed
    - the distance the last to-be-passed object will travel in the time to perform the maneuver
    - the distance of the closest object to the host vehicle.
- A notification is sent in the following cases:
  - there are any oncoming objects within the calculated maneuver zone
  - there are any oncoming objects that would reach the maneuver zone in less than the calculated passing time plus 2 s
  - the calculated full distance is larger than limitPassingDistance and limitPassingDistance is greater than 0
  - the calculated "length to be passed" is is greater than limitRelativeDistance and limitRelativeDistance is greater than 0
  - the number of cars to be passed in a single maneuver is greater than limitCarCount and limitCarCount is greater than 0.

4.3.3. DNP configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.dnp: true**
- The following parameters can be configured in **dnp** member
  - **sendPreventiveInfo**: Send a notification even if a turn signal is not used, but at INFO level instead of ALERT.
  - **additionalOvertakingSpeed**: Assumed additional speed for overtaking at same-speed situations [m/s].
  - **mergingTimeGap**: Minimum time gap between following vehicles for allowing merge [s].
  - **activationTimeGap**: Time gap between the host and remote vehicles ahead for application activation [s].
  - **activationDistance**: Distance between the host and remote vehicles ahead for application activation [m].
  - **limitPassingDistance**: Disallow overtaking above the set passing zone length, 0: off [m].
  - **limitRelativeDistance**: Disallow overtaking above the set relative distance, 0: off [m].
  - **limitCarCount**: Disallow overtaking above the set car count ahead, 0: off.

Common configuration parameters affecting the application in **cff.json**:

- [Lane abstraction] configuration in **core.classification.lane.width** and **.overlap** member
- [Same direction] classifier configuration in **core.classification.headingDiffThr** member
- [Opposite direction] classifier configuration in **core.classification.headingDiffThr** member

4.3.4. DNP Notifications

This application sends ALERT level notifications, which contain the following data:

- The Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)).
- The Object Data and Relative Data of the nearest ahead object.
- The follow time to the nearest object (the gap divided by the speed of the host vehicle).

- The Object Data and Relative Data of the nearest relevant oncoming object.
- The passing time of the nearest relevant oncoming object (the distance to it divided by its relative speed).
- The parameters of the passing maneuver:
  - The number of cars to be passed.
  - The full traveled distance.
  - The relative distance to be passed.
  - The calculated passing time plus 2 s.

4.4. Emergency Electronic Brake Light (EEBL)

This safety application issues a warning about a hard breaking vehicle ahead.

4.4.1. EEBL safety application overview

The Emergency Electronic Brake Light (EEBL) safety application is triggered by a hard braking event in front of the host vehicle, even if the braking vehicle is obstructed from view (for example by a larger vehicle or weather conditions).

This application is compliant with the European (ETSI G5) and the US (Wave) standards, it issues the same trigger to the vehicle driver. However the Emergency Electronic Brake Light applications internal logic and function is different for the two regions as described below.

EEBL

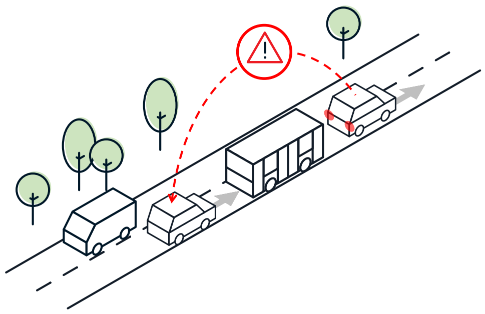

Figure 5. EEBL Safety Application

4.4.2. EEBL mode of operation

Model

- EEBL is a V2V application, that works for the **Vehicle** remote entity type.
- The application is enabled or disabled based on the speed of the host vehicle (for more information, see “Speed-dependent behavior” (p. 3)).
- This array can be sent to the UDP adapter.
- EEBL uses [lane abstraction] for the target area.

Filter setup

- The remote vehicle is [within one lane] in the range of **targetLength** ahead. An option to adapt target length for the host speed is also available (**useAdaptiveTarget**).
- The remote vehicle is traveling in the [same direction].
- The remote vehicle is located [ahead].
- The remote vehicle is located at the [same altitude].

Triggering conditions

- The application triggers when a hard braking event occurs, which is determined by:
  - [hard braking flag (HBF)] and [longitudinal acceleration] of the remote vehicle (**useHbf: true**)

  - only [longitudinal acceleration] of the remote vehicle in case of (**useHBF: false**)
- The data sources of the hard braking event are:
  - [HBF]: remote vehicle is broadcasting a hard braking event and the application is configured to check it. If the application is not configured to use the HBF, then this is not checked.
  - [acceleration:] (which should be negative in this case) is less than the configured threshold (**accelerationThr**). Note: "less" indicates a higher absolute value as the limit is negative.

4.4.3. EEBL configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.eebl: true**
- The following parameters can be configured in the **eebl** key
  - **targetLength**: The base length of the target area [m]
  - **useAdaptiveTarget**: Adapt effective **targetLength** in proportion to the host vehicle's speed
  - **accelerationThr**: Threshold of the acceleration of the remote vehicle (a negative value is required, because in this case the remote vehicle is decelerating) for detecting a hard braking event, in [m/s^2]
  - **useHbf**: Use **HBF** data state "ON" of remote vehicle for trigger condition.

Common configuration parameters affecting the application in **cff.json**:

- [Lane abstraction] configuration in **core.classification.lane.width** and **.overlap**
- [Same direction] classifier configuration in **core.classification.headingDiffThr**
- [Same altitude] classifier configuration in **core.classification.altitudeDiffThr**

4.4.4. EEBL notifications

The application sends ALERT level notifications.

The notification contains the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) and the Object Data and Relative Data of the remote vehicle.

4.5. Emergency Vehicle Warning (EVW)

This safety application issues a warning about an approaching emergency vehicle.

4.5.1. EVW safety application overview

The Emergency Vehicle Warning (EVW) safety application is a V2V communication based safety feature that issues a warning to the driver of the host vehicle in case of an approaching emergency vehicle, either in the same path of travel or crossing an intersection ahead.

The EVW helps drivers in recognizing traffic situations where right of way must be given to an emergency vehicle, even if the emergency vehicle is obstructed from view. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

EVW

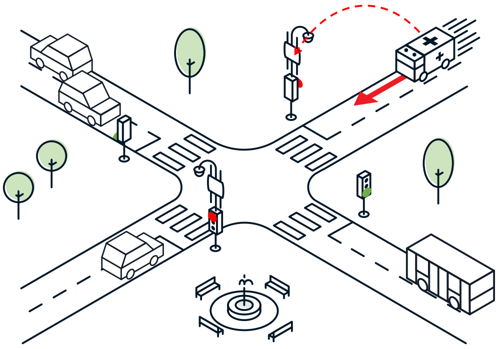

Figure 6. EVW Safety Application

4.5.2. EVW mode of operation

Model

- The Emergency Vehicle Warning (EVW) safety application sends a notification to the driver if a probably non-visible emergency vehicle with sirens on is approaching.
- The application uses the [Object] entity type.
- If enabled, the **targetRadius** value can be changed based on the speed of the host vehicle: The effective **targetRadius** is equal to **targetRadius** × (host speed [km/h]/50 [km/h]). This value is fixed to be in the range of 50–1000 m.

Filter setup

The application identifies vehicles with matching the following criteria:

- The remote vehicle is within **targetRadius** distance.
- The remote vehicle has a "siren" flag set to [ON].

Triggering conditions

A notification is sent on every processing cycle, for every filtered vehicle, if any of the following conditions are true:

- The remote vehicle is [within one lane], [behind] and heading in the [same direction] relative to the host vehicle.
- The remote vehicle is [ahead] of and [approaching from the side], towards the host vehicle.
- It is coming towards the host at a 45° angle:
  - It is ahead and to the right of the host vehicle, and it's relative heading is within 112.5° and 157.5°
  - It is ahead and to the left of the host vehicle, and it's relative heading is within −157.5° and −112.5°
  - It is behind and to the right of the host vehicle, and it's relative heading is within −67.5° and −22.5°
  - It is behind and to the left of the host vehicle, and it's relative heading is within 22.5° and 67.5°.

4.5.3. EVW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.evw: true**
- The following parameters can be configured in the **evw** key:
  - **targetRadius**: radius of the circular target area in meters [m]
  - **useAdaptiveTarget**: Use speed-dependent target radius scaling
  - If enabled, the **targetRadius** value changes based on the speed of the host vehicle: The effective **targetRadius** is equal to **targetRadius** × (host speed [km/h]/50 [km/h]). This value is fixed to be in the range of 50–1000 m.

4.5.4. EVW notifications

The application sends ALERT level notifications which contain the Object Data of the host vehicle and the Object Data and Relative Data of the remote vehicle (for more information, see “Common notification contents” (p. 2)).

4.6. Forward Collision Warning (FCW)

This safety application issues a warning about an impending rear-end collision.

4.6.1. FCW safety application overview

The Forward Collision Warning (FCW) safety application is a V2V communication based safety feature that issues a warning to the driver of the host vehicle in case of an impending rear-end collision with a remote vehicle ahead in traffic in the same lane and direction of travel.

The FCW helps drivers in avoiding or mitigating front-to-rear vehicle collision in the forward path of travel. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

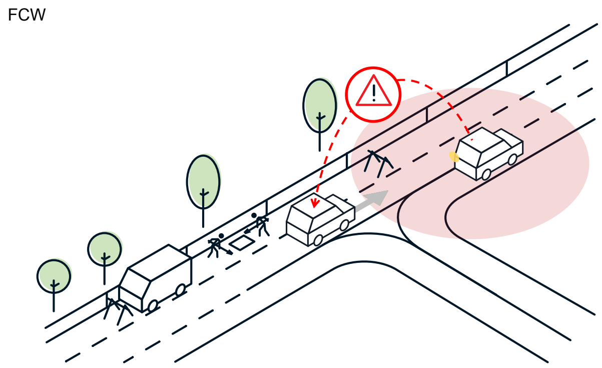

Figure 7. FCW Safety Application

4.6.2. FCW mode of operation

Model	• The FCW is a V2V application, that works for the **Object** remote entity type. • FCW is a V2V safety application that is enabled or disabled based on the speed of the host vehicle (for more information, see “Speed-dependent behavior” (p. 3)). • The application uses [lane abstraction] for target area. • The application is triggered based on [time-to-collision (TTC)].
Filter setup	• The remote vehicle is in the [same lane] in the range of **target-Length** ahead. Optionally the target length can be adapted for the speed of the host vehicle (**useAdaptiveTarget**) • The remote vehicle is going the [same direction] • The remote vehicle is [ahead] • The remote vehicle is at the [same altitude].
Triggering conditions (normal mode)	If multi-object mode is not enabled, on every processing cycle, for every filtered object: • The application computes the time it would take for the host vehicle to collide with the remote object, assuming the following: • Both stay in the same lane and travel in a straight line, in the same direction. • Both are either moving with constant velocity or with constant acceleration, depending on the configuration.

	• The gap to close between the two is the same as the distance of their center points minus half of both of their lengths. • If this time-to-collision is less than the TTC threshold, a notification is sent.
Triggering conditions (multi-object mode)	If multi-object mode is enabled, the following calculations are handled on every processing cycle: • The objects in front of the host vehicle are accumulated and sorted from closes to farthest. Only the closest 32 objects are kept. • If there is a predicted collision with the closest object according to the normal mode rules, a notification is sent. • If there is a predicted collision between any consecutive vehicles in the string (according to normal rules, but between two remote vehicles), the following takes place: • The application computes when the crash happens. • Assuming that the crashed cars stop instantly and the rest of the cars crash into them and stop instantly, it computes a the distance to the full wreckage, using the pre-crash gap lengths. • Assuming constant velocity, computes the additional TTC. • If the sum of the TTC of the original crash and then to reach the wreckage is less than the TTC threshold, a notification is sent.

4.6.3. FCW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.fcw: true**
- The following parameters can be configured in the **fcw** key
  - **targetLength**: The base length of the target area [m]
  - **useAdaptiveTarget**: Adapt effective **targetLength** in proportion to the speed of the host vehicle
  - **TTC**: Time-to-collision threshold, in [s]
  - **useAdaptiveTtc**: Use speed-dependent TTC scaling. It is recommended to set either this or **useAccelerationForTtc** to true.
  - **useAccelerationForTtc**: Assume constant acceleration when calculating the minimum distance point. It is recommended to set either this or **useAdaptiveTtc** to true.

Common configuration parameters affecting the application in **cff.json**:

- [Lane abstraction] configuration in **core.classification.lane.width** and **.overlap**
- [Same direction] classifier configuration in **core.classification.headingDiffThr**
- [Same altitude] classifier configuration in **core.classification.altitudeDiffThr**

4.6.4. FCW Notifications

The application sends ALERT level notifications which contain the Object Data of the host vehicle and the Object Data and Relative Data of the remote vehicle (for more information, see “Common notification contents” (p. 2)). The notification data also contains [Time-to-collision], the IDs of the originally colliding objects, and the TTC of the original collision. These field are only valid in multi-object mode.

4.7. Intersection Movement Assist (IMA)

This safety application issues a warning about an impending collision with a vehicle approaching the same intersection in a perpendicular direction.

4.7.1. IMA safety application overview

The Intersection Movement Assist (IMA) is a V2V communication based safety application that issues a warning to the driver of the host vehicle in case of an impending side-to-front/front-to-side collision with a remote vehicle driving in a direction that is perpendicular to the host vehicle, typically in an intersection.

The IMA 0helps drivers in avoiding or mitigating side-to-front or front-to-side vehicle collision in the forward path of travel. This application is compliant and relies on the European (ETSI G5) and the US (WAVE) standards.

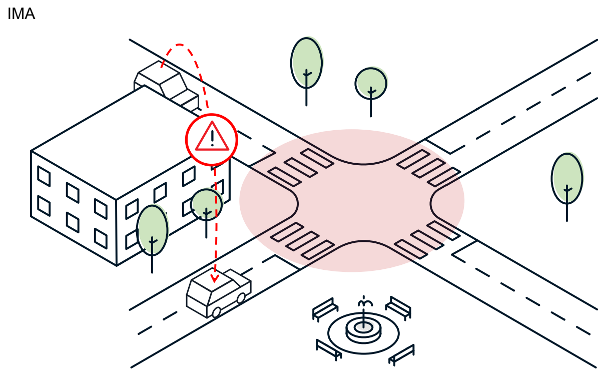

Figure 8. IMA Safety Application

4.7.2. IMA mode of operation

Model

- The IMA is a V2V application, that works for the **Object** remote entity type
- IMA is enabled or disabled based on the speed of the host vehicle (for more information, see “Speed-dependent behavior” (p. 3)).
- If enabled, the **targetLength** value can be changed based on the speed of the ego vehicle. The effective **targetLength** is equal to **targetLength** × (host speed [km/h]/50 [km/h]). This value must be in the range of 10–1000 m.
- If enabled, the maximum TTC value changes based on the speed of the ego vehicle. The effective maximum TTC is equal to maximum TTC× (host speed [km/h]/50 [km/h]). If this computed effective value is less than 3 s,then it is set to be equal to 3 s.
- The application uses [distance based filtering] for target area.
- The application is triggered based on the minimum predicted distance of objects.
- Precondition for the application: the remote object is not stationary. The application uses a predefined speed limit to decide whether or not the object is stationary.

Filter setup

- The remote vehicle is in the [range] of **targetRadius**. This can optionally be set to adapt target radius for host speed, using **useAdaptiveTarget**
- The remote vehicle is approaching [from side direction]
- The remote vehicle is located [ahead]
- The remote vehicle is in the [same altitude]
- The remote vehicle is moving with at least 5 km/h.

Triggering conditions

On every processing cycle, for every filtered object:

- The application computes the closest it will get to the object, assuming both are moving on a straight line, at either constant speed or constant acceleration depending on the configuration.
- The calculated closest distance is corrected using the length of both objects.
- If this distance is less than the configured minimum clearance distance and the time it will take to reach this distance is below the configured maximum TTC, a notification is sent

4.7.3. IMA configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.ima: true**
- The following parameters can be configured in the **ima** member
  - **targetRadius**: The base radius of the target area in [m]
  - **useAdaptiveTarget**: Adapt effective **targetRadius** in proportion to the host vehicle's speed
  - **dangerousDistance**: The distance of the remote vehicle to be considered [dangerous], in [m]
  - **TTC**: Time-to-collision threshold, in [s]
  - **useAdaptiveTtc**: Adapt effective **TTC** proportionally to the speed of the host vehicle
  - **useAccelerationForTtc**: Use host and remote vehicle acceleration data when calculating **TTC**

Common configuration parameters affecting the application in **cff.json**:

- [From side] classifier configuration in **core.classification.headingDiffThr**
- [Same altitude] classifier configuration in **core.classification.altitudeDiffThr**

4.7.4. IMA Notifications

The application sends ALERT level notifications which contain the Object Data of the host vehicle and the Object Data and Relative Data of the remote vehicle (for more information, see “Common notification contents” (p. 2)).

The application also sends Time to Collision information in the notification as well. This is the predicted time (in ms) until the two vehicles are closest to each other.

4.8. Lane Change Assist (LCA)

This safety application issues a warning about an impending collision during lane changing.

4.8.1. LCA safety application overview

The Lane Change Assist (LCA) application assists a vehicle that is about to change lanes. The warning is provided in order to avoid a collision with remote vehicles in the intended lane destination of the host vehicle. The application operates using a target area and various filtering and triggering criteria as well. The application is active if the speed of the host vehicle is within the specified range.

This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

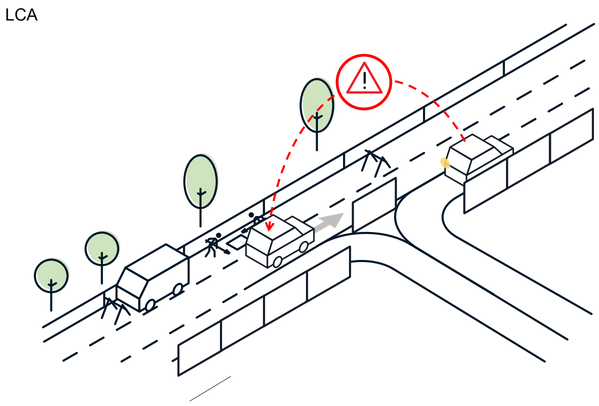

Figure 9. LCA Safety Application

4.8.2. LCA mode of operation

Model

- LCA is a V2V safety application that is enabled or disabled based on the speed of the host vehicle (for more information, see “Speed-dependent behavior” (p. 3)).
- If enabled, the **targetLength** value can be changed based on the speed of the host vehicle. The effective **targetLength** is equal to **targetLength** × (host speed [km/h]/50 [km/h]). This value must be in the range of 10–1000 m.
- If enabled, the maximum TTC value changes based on the speed of the host vehicle. The effective maximum TTC is equal to maximum TTC × (host speed [km/h]/50 [km/h]). If this computed effective value is less than 3 s, then it is set to be equal to 3 s.
- The application is enabled or disabled based on the speed of the host vehicle.
- The application uses [lane abstraction] for target area.
- The application is triggered based on the turn signal status of the host vehicle, the [time to collision (TTC)], and the relative distance of the remote vehicle.

Filter setup

- The remote vehicle is in the [one lane left] or [one lane right] position when the corresponding turn signal is active on the host vehicle. [One lane left] if only the left turn signal is active or [one lane right] if only the left signal is active. If signal state is unknown, both signals are off, or both are on, then no objects will be filtered, effectively disabling the application.
- The remote vehicle is at least **targetOffset** meters behind (or ahead, if **targetOffset** is negative)
- The remote vehicle is at most **targetOffset** + **targetLength** behind
- The remote vehicle is going the [same direction] as the host vehicle
- The remote vehicle is at the [same altitude] as the host vehicle.

Triggering conditions

On every processing cycle, for every filtered object:

- The application computes the time it would take for the host vehicle to collide with the remote object, assuming the following:
  - The host vehicle moves into the lane of the remote vehicle and then both stay in the same lane, and travel in a straight line, in the same direction.
  - Both are either moving with constant velocity or with constant acceleration, depending on the configuration.
  - The gap to close between the two is the same as the distance of their center points minus half of both of their lengths.
- A notification is sent, if the following criteria are met:
  - The time to collision is less than the TTC threshold.
  - The object is within **dangerousDistance** (the distance between the center point of the host vehicle and the object is less than **dangerousDistance**).

4.8.3. LCA configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.lca: true**
- The following parameters can be configured in the **lca** key
  - **targetLength**: The base length of the target area in [m]
  - **targetOffset**: Offset of the remote vehicle's front from the reference point of the host vehicle [m]
  - **useAdaptiveTarget**: Adapt effective **targetRadius** in proportion to the speed of the host vehicle

- **dangerousDistance**: The distance of the remote vehicle to be considered [dangerous], in [m]
- **TTC**: Time-to-collision threshold, in [s]
- **useAdaptiveTtc**: Use speed-dependent TTC scaling. It is recommended to set either this or **useAccelerationForTtc** to true.
- **useAccelerationForTtc**: Assume constant acceleration when calculating the minimum distance point. It is recommended to set either this or **useAdaptiveTtc** to true.

Common configuration parameters affecting the application in **cff.json**:

- [Lane abstraction] configuration in **core.classification.lane.width** and **.overlap**
- [Same direction] classifier configuration in **core.classification.headingDiffThr**
- [Same altitude] classifier configuration in **core.classification.altitudeDiffThr**

4.8.4. LCA Notifications

The application sends ALERT level notifications which contain the Object Data of the host vehicle and the Object Data and Relative Data of the remote vehicle (for more information, see “Common notification contents” (p. 2)).

The application also sends Time-to-Collision information in the notification as well. This is the predicted time (in ms) until the two vehicles are closest to each other.

4.9. Left Turn Assist (LTA)

This safety application warns the driver about the possible dangers of a left turn maneuver in an intersection.

4.9.1. LTA safety application overview

Turn Assist applications offer insight to the driver of the host vehicle in a turning traffic scenario. In the left turn use case, the remote vehicle approaches from the opposite direction, crossing the intended path of the host vehicle.

This application and the configuration parameters detailed in this chapter are optimized for right-hand traffic. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

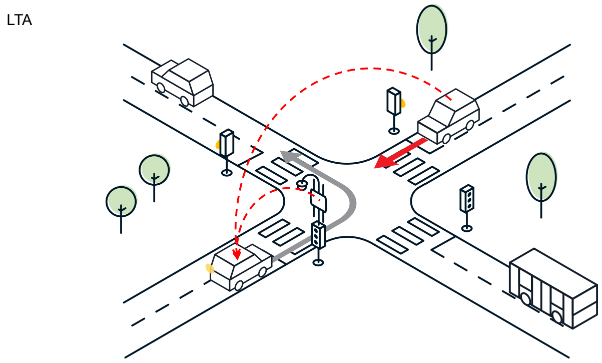

Figure 10. LTA Safety Application

4.9.2. LTA mode of operation

Model	• The LTA is a V2V application, that works for the **Object** remote entity type. • LTA is enabled or disabled based on the host vehicle's speed (for more information, see “Speed-dependent behavior” (p. 3)). • The application uses [polygon] for target area on the left.

- The application is triggered based on the time required to finish the turning maneuver.
- Precondition: the left turn signal of the host vehicle is turned on.

Filter setup

- The remote vehicle inside the rectangular target area is defined by the following criteria:
  - The center of the host vehicle
  - **targetLength** distance in front of the host vehicle
  - **turnRadius** distance to the left of the host vehicle
  - **targetLength** in front and **turnRadius** to the left of the host vehicle.

  The target area also takes the dimensions of the remote vehicle into account if available.
- The remote vehicle is going the [opposite direction]
- The remote vehicle is in the [same altitude]

Triggering conditions

If the host is not signaling neither to the left nor to the right, no notifications will be sent. Otherwise on every processing cycle, for every filtered object, the following calculations are handled:

- The application computes the time it will take for the host vehicle to take the turn. It is the sum of:
  - The time it takes to decelerate to **turnSpeed**. This is computed as (**turnSpeed** − actual host speed) **timeCorrectionSpeedCoeff**
  - The time it takes to do the turn. This is computed as **turnRadius** × π/(2 × **turnSpeed**)
- The application computes the time it takes for the remote vehicle to reach the end point of the turn:
  - The distance it has to travel is computed as the difference between the relative longitudinal distance and the turn radius.
  - The time is this distance divided by the speed of the remote vehicle.
- The application is triggered if the remote vehicle reaches the [area of turn] within the estimated time frame of the turn maneuver. The turn is considered dangerous if time difference is below **safetyTime**.

4.9.3. LTA configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.lta: true**
- The following parameters can be configured in the **lta** key
  - **vStat**: Speed threshold when the vehicle is considered stationary [m/s]
  - **targetLength**: The base length of the target area in [m]
  - **turnRadius**: the estimated radius of the predicted left turn in [m]
  - **turn**Speed: the estimated speed of the predicted left turn in [m/s]
  - **safetyTime**: Time margin for a safe turn in [s]
  - **timeCorrectionSpeedCoeff**: Coefficient of compensating maneuver time based on host speed

Common configuration parameters affecting the application in **cff.json**:

- [Opposite direction] configuration in **core.classification.headingDiffThr** member
- [Same altitude] classifier configuration in **core.classification.altitudeDiffThr** member

4.9.4. LTA Notifications

The application sends ALERT level notifications which contain the Object Data of the host vehicle and the Object Data and Relative Data of the remote vehicle (for more information, see “Common notification contents” (p. 2)).

The application also sends Time-to-Collision information in the notification as well. This is the predicted time (in ms) until the two vehicles are closest to each other.

4.10. Pedestrian Collision Warning (PCW)

This safety application warns the driver of the host vehicle about a potentially dangerous collision with a pedestrian in the direction of traffic.

4.10.1. PCW safety application overview

The Pedestrian Collision Warning (PCW) safety application is a V2I communication based safety feature that issues a warning to the driver if there is an object ahead in the same lane that is classified as pedestrian. This application is triggered based on the lane information of the surrounding area, the path information of the vehicle, and the location of the pedestrian. Pedestrian information (for example their location, their classification as pedestrian, their projected path of travel) can be received from nearby sensors or mobile devices carried by the pedestrians.

The PCW helps drivers to avoid or mitigate the risk of accidents involving vehicle-to-pedestrian collisions, increasing safety, and reducing traffic congestion. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

PCW

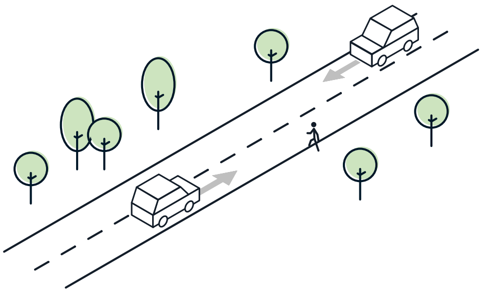

Figure 11. PCW Safety Application

4.10.2. PCW mode of operation

Model

- PCW is a V2V safety application that is enabled or disabled based on the host vehicle's speed(for more information, see “Speed-dependent behavior” (p. 3)).
- If configured to do so, the **targetLength** value changes based on the speed of the ego vehicle. The effective **targetLength** will be equal to **targetLength** × (host speed [km/h]/50 [km/h]). This value must be in the range of 10–1000 m.
- If enabled, the maximum TTC value can be changed based on the speed of the host vehicle. The effective maximum TTC is equal to maximum TTC× (host speed [km/h]/50 [km/h]. If this computed effective value is less than 3 s, then it is set to be equal to 3 s.
- The application uses the [Object] entity type.

Filter setup

The application subscribes to remote [Objects] which match the following criteria:

- The remote [Object] is [Pedestrian] type.
- The pedestrian is at most **targetLength** distance ahead.

Triggering conditions (normal mode)

If simple mode is not selected, on every processing cycle, for every filtered object:

- The application computes the closest it will get to the pedestrian, assuming both are moving on a straight line, at either constant speed or constant acceleration depending on the configuration.

	• If this distance is less than the configured minimum clearance distance and the time it will take to reach this distance is below the configured maximum TTC, a notification is sent.
Triggering conditions (simplified mode)	If simple mode is selected, on every processing cycle, for every filtered object: • The application computes the time it would take for the host vehicle to collide with the pedestrian, assuming the following: • The pedestrian stays stationary (even if the received data indicates nonzero speed or acceleration). • The host vehicle continues to move in a straight line with constant velocity. • If the computed TTC is less than the configured threshold, a notification is sent.

4.10.3. PCW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.pcw: true**
- The following parameters can be configured in the **pcw** key:
  - **targetLength**: Length of the target area in [m]
  - **useAdaptiveTarget**: Use speed-dependent target length scaling.
  - **dangerousDistance**: The minimum clearance distance that is considered safe.
  - **ttc**: The maximum time it can take to reach the dangerous distance.
  - **useAdaptiveTtc**: Use speed-dependent TTC scaling. It is recommended to set either this or **useAccelerationForTtc** to true.
  - **useAccelerationForTtc**: Assume constant acceleration when calculating the minimum distance point. It is recommended to set either this or **useAdaptiveTtc** to true.
  - **useSimpleAlgorithm**: Use a simplified algorithm that assumes all pedestrians are stationary.

4.10.4. PCW notifications

The application sends ALERT level notifications which contain the Object Data of the host vehicle and the Object Data and Relative Data of the remote vehicle (for more information, see “Common notification contents” (p. 2)).

The application also sends Time to Collision information in the notification as well. This is the predicted time (in ms) until the two vehicles are closest to each other.

4.11. Right Turn Assist (RTA)

This safety application warns the driver about the possible dangers of a right turn maneuver in an intersection.

4.11.1. RTA safety application overview

Turn Assist applications offer insight to the driver of the host vehicle in a turning traffic scenario. In the right turn use case, the remote vehicle approaches from a perpendicular direction, crossing the intended path of the host vehicle.

This application and the configuration parameters detailed in this chapter are optimized for right-hand traffic; however, the same application logic can be customized for left-hand traffic as well. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

RTA

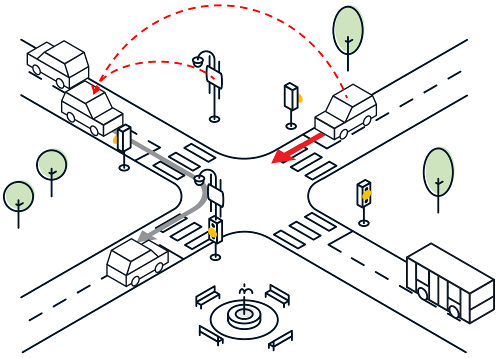

Figure 12. RTA Safety Application

4.11.2. RTA mode of operation

Model

- The RTA is a V2V application, that works for the **Object** remote entity type.
- CLW is enabled or disabled based on the speed of the host vehicle (for more information, see “Speed-dependent behavior” (p. 3)).
- The application uses the [polygon] classification as the target area in the perpendicular direction on the right.
- The application is triggered based on the time required to finish the turning maneuver
- Precondition: the right turn signal of the host vehicle is turned on.

Filter setup

- The remote vehicle is inside rectangular target area defined by the following criteria:
  - The center of the host vehicle
  - **targetLength** distance in front of the host vehicle
  - **turnRadius** distance to the left of the host vehicle
  - **targetLength** in front and **turnRadius** to the left of the host vehicle

  The target area also takes the dimensions of the remote vehicle into account if available.
- The remote vehicle is traveling in the [opposite direction]
- The remote vehicle is in the [same altitude]

Triggering conditions

If the host is not signaling neither to the left nor to the right, no notifications will be sent. Otherwise on every processing cycle, for every filtered object, the following calculations are handled:

- The application computes the time it will take for the host vehicle to take the turn. It is the sum of:
  - The time it takes to decelerate to **turnSpeed**. This is computed as (**turnSpeed** − actual host speed) × **timeCorrectionSpeedCoeff**
  - The time it takes to do the turn. This is computed as **turnRadius** × π/(2 × **turnSpeed**)
- The application computes the time it takes for the remote vehicle to reach the end point of the turn:
  - The distance it has to travel is computed as the difference between the relative longitudinal distance and the turn radius.
  - The time is this distance divided by the speed of the remote vehicle.
- The application is triggered if the remote vehicle reaches the [area of turn] within the estimated time frame of the turn maneuver. The turn is considered dangerous if time difference is below **safetyTime**.

4.11.3. RTA configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.rta: true**
- The following parameters can be configured in the **rta** key
  - **vStat**: Speed threshold when the vehicle is considered stationary [m/s]
  - **targetLength**: The base length of the target area in [m]
  - **checkAheadDist**: Additional distance to define the width of the target area [m]
  - **turnRadius**: the estimated radius of the theoretical left turn in [m]
  - **turn**Speed: the estimated speed of the theoretical left turn in [m/s]
  - **safetyTime**: Time margin for a safe turn in [s]
  - **timeCorrectionSpeedCoeff**: Coefficient of compensating maneuver time based on host speed

Common configuration parameters affecting the application in **cff.json**:

- [From left] config in **core.classification.headingDiffThr** member
- [Same altitude] classifier config in **core.classification.altitudeDiffThr** member

4.11.4. RTA Notifications

The application sends ALERT level notifications which contain the Object Data of the host vehicle and the Object Data and Relative Data of the remote vehicle (for more information, see “Common notification contents” (p. 2)).

The application also sends Time to Collision information in the notification as well. This is the predicted time (in ms) until the two vehicles are closest to each other.

4.12. Stationary Vehicle Warning (SVW) - ASV use case

The Stationary Vehicle Warning (SVW) safety application, used in Abnormal Stationary Vehicle (ASV) mode sends a notification to the driver when there is a stationary vehicle with hazard lights on in front of the host vehicle.

4.12.1. SVW safety application overview

The Stationary Vehicle Warning (SVW) safety application is a V2V communication based safety feature that issues a warning to the driver of the host vehicle in case of a remote vehicle is stationary ahead in traffic in the same lane and direction of travel—even if the remote vehicle is obstructed from view.

The SVW helps drivers in avoiding or mitigating front-to-rear vehicle collision in the forward path of travel with a stationary vehicle. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

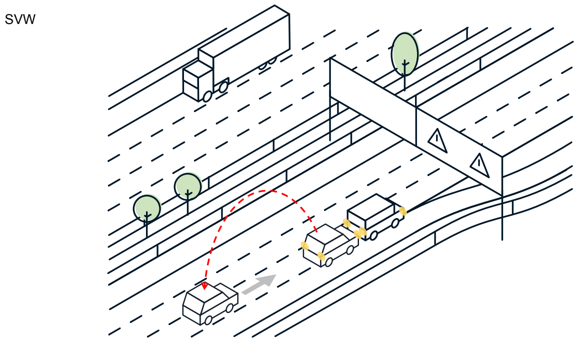

Figure 13. SVW Safety Application

4.12.2. SVW mode of operation

Model	• SVW is a V2V application, that works for the **Vehicle** remote entity type. • The application is enabled or disabled based on the host vehicle's speed (for more information, see “Speed-dependent behavior” (p. 3)). • If enabled, the **targetLength** value can be changed based on the speed of the host vehicle. The effective **targetLength** is equal to **targetLength***(host speed / 50km/h). This value must be between 10 and 1000 meters. • SVW uses [lane abstraction] for the target area.
Filter setup	• The remote vehicle is [within one lane] in the range of **targetLength** ahead. There's also an option available to adapt target length for host speed (**useAdaptiveTarget**). • The remote vehicle is considered stationary, so their speed is at most **stationarySpeedThr**. • The remote vehicle has its hazard lights turned ON. • The remote vehicle is located [ahead] • The remote vehicle is located at the [same altitude]
Triggering conditions	The application triggers a notification every processing cycle for every filtered object it receives.

4.12.3. SVW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.svw: true AND svw.triggerForAbnormalVehicles: true**
- The following parameters can be configured in the **svw** key
  - ASV use case specific settings can be configured under the **asv** key
    - **targetLength**: The base length of the target area [m]
    - **useAdaptiveTarget**: Adapt effective **targetLength** in proportion to the host vehicle's speed
    - **stationarySpeedThr**: The speed threshold, below which a remote object is considered stationary.

Common configuration parameters affecting the application in **cff.json**:

- [Lane abstraction] configuration in **core.classification.lane.width** and **.overlap**
- [Same direction] classifier configuration in **core.classification.headingDiffThr**
- [Same altitude] classifier configuration in **core.classification.altitudeDiffThr**

4.12.4. SVW notifications

The application sends INFO level notifications.

The notification contains the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) and the Object Data and Relative Data of the remote vehicle.

The application also sends Time to Collision information in the notification as well. This is the predicted time (in milliseconds) until the two vehicles are closest to each other.

4.13. Stationary Vehicle Warning (SVW) - TSV use case

The Stationary Vehicle Warning (SVW) safety application, used in Target-based Stationary Vehicle use-case (TSV) mode sends a notification to the driver of the host vehicle in case of an impending front-end collision with a remote stationary vehicle ahead in the same lane.

4.13.1. SVW safety application overview

The Stationary Vehicle Warning (SVW) safety application is a V2V communication-based safety feature that issues a warning to the driver of the host vehicle in case of a remote vehicle is stationary ahead in traffic in the same lane and direction of travel - even if the remote vehicle is obstructed from view.

SVW will help drivers in avoiding or mitigating front-to-rear vehicle collision in the forward path of travel with a stationary vehicle. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

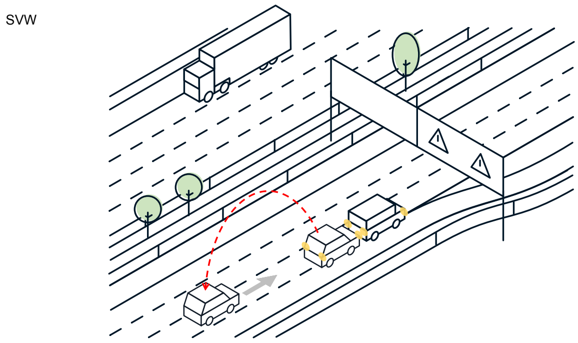

Figure 14. SVW Safety Application

4.13.2. SVW mode of operation

Model

- SVW is a V2V application, that works for the **Vehicle** remote entity type.
- The application is enabled or disabled based on the host vehicle's speed (for more information, see “Speed-dependent behavior” (p. 3)).
- If configured to do so, the **targetLength** value changes based on the speed of the ego vehicle. The effective **targetLength** will be equal to **targetLength** × (host speed [km/h]/50 [km/h]). This value must be in the range of 10–1000 m.
- If configured to do so, the maximum TTC value changes based on the speed of the ego vehicle. The effective maximum TTC is equal to maximum TTC × (host speed [km/h]/50 [km/h]). If this computed effective value is less than 3 s, then it is set to be equal to 3 s.
- SVW uses [lane abstraction] for the target area.

Filter setup

- The remote vehicle is [within one lane] in the range of **targetLength** ahead. An option to adapt target length for the speed of the host vehicle is also available (**useAdaptiveTarget**).
- The remote vehicle is considered stationary; thus, their speed is at most **stationarySpeedThr**.
- The remote vehicle has its hazard lights turned ON.
- The remote vehicle is located [ahead].
- The remote vehicle is located at the [same altitude].

Triggering conditions

The following calculations are handled on every processing cycle, for every filtered object:

- The application computes the time it would take for the host vehicle to collide with the remote object, assuming the following:
  - Both stay in the same lane and travel in a straight line, in the same direction.
  - Both are either moving with constant velocity or with constant acceleration, depending on the configuration.

    - The gap to close between the two is the same as the distance of their center points minus half of both of their lengths.
  - If this time to collision is less than the TTC threshold, a notification is sent.

4.13.3. SVW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.svw: true AND svw.triggerForAbnormalVehicles: true**
- The following parameters can be configured in the **svw** key
  - TSV use case specific settings can be configured under the **tsv** key
    - **targetLength**: The base length of the target area [m]
    - **useAdaptiveTarget**: Adapt effective **targetLength** in proportion to the speed of the host vehicle
    - **ttc**: Time to collision threshold
    - **useAdaptiveTtc**: Use speed-dependent TTC scaling. It is recommended to set either this or **useAccelerationForTtc** to true.
    - **useAccelerationForTtc**: Assume constant acceleration when calculating the minimum distance point. It is recommended to set either this or **useAdaptiveTtc** to true.
    - **stationarySpeedThr**: The speed threshold, below which a remote object is considered stationary.

Common configuration parameters affecting the application in **cff.json**:

- [Lane abstraction] configuration in **core.classification.lane.width** and **.overlap**
- [Same direction] classifier configuration in **core.classification.headingDiffThr**
- [Same altitude] classifier configuration in **core.classification.altitudeDiffThr**

4.13.4. SVW notifications

The application sends ALERT level notifications.

The notification contains the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) and the Object Data and Relative Data of the remote vehicle.

The application also sends Time to Collision information in the notification as well. This is the predicted time (in ms) until the two vehicles are closest to each other.

5. Infrastructure-to-Vehicle (I2V) applications

The following sections contain information about safety applications and their various use cases that rely on V2X communication between the infrastructure and vehicles (also referred to as V2I applications in some context).

5.1. Traffic Information Warning (TIW) - Speed Limit Excess Warning (SPD) use case

The Speed Limit Excess Warning (SPD) use case warns the driver about exceeding a speed limit.

5.1.1. SPD safety application overview

The Speed Limit Warning (SPD) is a use case for the Traffic Information Warning (TIW) I2V safety application that provides a notification message to the driver of the host vehicle, received from the road infrastructure about applicable speed limitations valid for the given road segment (normally conveyed through static or variable message signs along the road).

The SPD helps drivers to receive information and reminders about speed limits, reducing excessive speeding in traffic. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

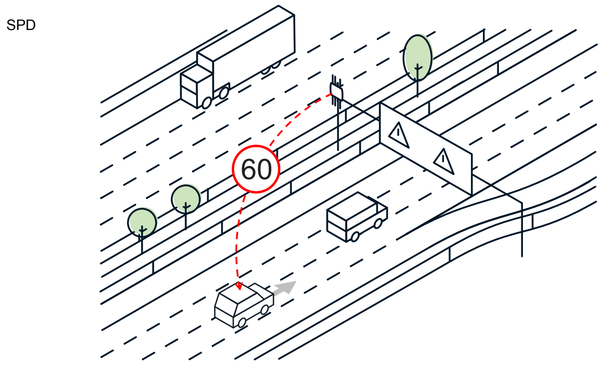

Figure 15. SPD Safety Application

5.1.2. SPD mode of operation

Model

- The SPD is a V2I application use case that works for the **TrafficInfo** remote entity type.
- The application is enabled or disabled based on the speed of the host vehicle (for more information, see “Speed-dependent behavior” (p. 3)).
- The application uses object-in-area algorithms for filtering.
- The application's triggering conditions are based on [ITIS codes]and the configured speeding threshold.
- The application displays different notification levels based on the [TrafficInfo] type and the speeding threshold. The different notification levels are the following:
  - [Alert] - this notification is displayed when the host vehicle is speeding within a [Regulation] or [Danger] type.
  - [Warning] - this notification is displayed when the host vehicle is speeding within any other type, for example [Advice].
  - [Info] - this is an informative notification, displayed when the host vehicle is not speeding.

Filter setup

The application subscribes to Road Information, where:

- The host vehicle must be in the [relevance zone] of the **TrafficInfo** entity type. The relevance zone can be either a designated track or a circular area.
- The host vehicle must also be at the [same altitude] as the [relevance zone] of the TrafficInfo entity type.
- The heading of the host vehicle must be [aligned with the zone's direction]. In case of circular target areas the the direction is considered aligned if the host vehicle is moving toward/from the center of the area.

Triggering conditions

- The application is triggered based on ITIS codes that contain speed limit information in the following format: [SpeedLimit] + [N0-N255] + [mph] or [kph] (**[268, 12544-12799, 8720/8721]**)
- The application determines whether or not the host vehicle is speeding. The effective speed limit can be increased by a set percentage using **relativeOverspeedThr** and the minimum speeding threshold is set by **minOverspeedThr**
- If TIW is triggered by a filtered object with Road Information, but none of the use cases match any triggering criteria, then a general TIW notification is triggered. Generic TIW triggering must be explicitly enabled in the configuration of the application.

5.1.3. SPD configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.spd: true**
- The following parameters can be configured in **spd** member
  - **minOverspeedThr**: The set minimum for the additional speed for effective speed limit threshold in [m/s]
  - **relativeOverspeedThr**: Relative additional speed for effective speed limit threshold, set in percentages [%]

5.1.4. SPD and general TIW notifications

If the speed of the host vehicle is above the adjusted speed limit (which is equal to the actual speed limit × **relativeOverspeedThr**, or **minOverspeedThr**, whichever is larger), and the Road Information Type is Danger or Regulation, the level is ALERT.

If the speed of the host vehicle is above the adjusted speed limit, and the Road Information Type is other than Danger or Regulation, the level is WARNING.

Otherwise in SPD cases the notification level is INFO.

The notification contains the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) and the Object Data and Relative Data of the remote vehicle.

The application also sends speed limit information (in m/s) in the notification as well, in the SPD use case.

General TIW notifications also contain the use case, if multiple cases are triggered then the latest one.

TIW notifications also contain the ITIS code list of the Road Information object.

5.2. Traffic Information Warning (TIW) - Wrong Way Road Warning (WWR) use case

The Wrong Way Road Warning (WWR) use case warns the driver about entering a lane the wrong way.

5.2.1. TIW - WWR safety application overview

The Wrong Way Road Warning (WWR) is a use case for the Traffic Information Warning (TIW) I2V safety application that provides a notification message to the driver of the host vehicle, received from the road infrastructure about entering the given road segment the wrong way (normally conveyed through static or variable message signs along the road).

The WWR helps drivers to receive information and reminders about wrong way entries, reducing congestion and hazard in traffic. This application is compliant and relies on the US (WAVE) standard.

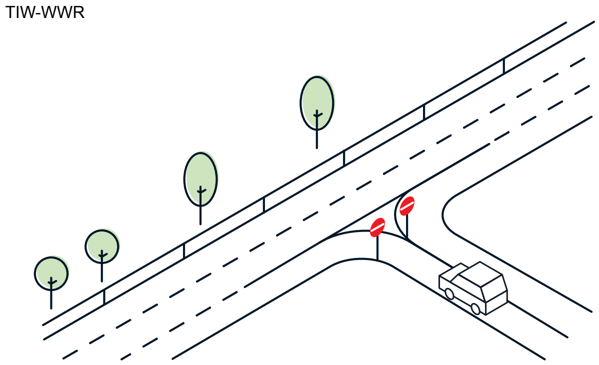

Figure 16. WWR Safety Application

5.2.2. TIW - WWR mode of operation

Model

- The WWR is a V2I application use case that works for the **TrafficInfo** remote entity type.
- The application uses object-in-area algorithms for filtering.
- The triggering conditions of the application are based on [ITIS codes] and the configured speeding threshold.

Filter setup

The TIW application subscribes to Road Information, where:

- The host vehicle must be in the [relevance zone] of the **TrafficInfo** entity type. The relevance zone can be either a designated track or a circular area.
- The host vehicle must also be at the [same altitude] as the [relevance zone] of the TrafficInfo entity type.
- The heading of the host vehicle must be [aligned with the zone's direction]. In the WWR use case the heading is the "wrong way" relative to the zone's direction.

Triggering conditions

- The application is triggered based on ITIS codes
- If the "Vehicle Travelling Wrong Way" ITIS code is found anywhere in the Road Information, then a WWR use case is triggered in TIW.
- If TIW is triggered by a filtered object with Road Information, but none of the use cases match any triggering criteria, then a general TIW notification is triggered.

5.2.3. TIW and WWR configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.tiw: true**
- The generic TIW application and the WWR use case have no additional configuration.

5.2.4. WWR and general TIW notifications

The TIW application displays different notification levels based on the [TrafficInfo] type and the speeding threshold. In the WWR use case, the notification level is always ALERT. The generic TIW different notification levels are the following:

- [Alert] - this notification is displayed when the host vehicle is speeding within a [Regulation] or [Danger] type.
- [Warning] - this notification is displayed when the host vehicle is speeding within any other type, for example [Advice].
- [Info] - this is an informative notification, displayed when the host vehicle is not speeding.

The notification contains the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) and the Object Data and Relative Data of the remote vehicle.

General TIW notifications also contain the use case, if multiple cases are triggered then the latest one.

TIW notifications also contain the ITIS code list of the Road Information object.

5.3. Traffic Information Warning (TIW) - Overheight Vehicle (OHV) use case

The Overheight Vehicle (OHV) warning informs the driver of the host vehicle about a section of the road ahead for which the host vehicle is considered overheight.

5.3.1. OHV safety application overview

The Overheight Vehicle (OHV) warning is a use case for the Traffic Information Warning (TIW) safety application which is a V2I communication based safety feature that alerts he driver of the host vehicle about an overpass or a tunnel ahead that is too low for their vehicle. This information is based on the received configured vehicle height and topography information from nearby RSUs and the current lane information of the host vehicle.

The OHV helps to prevent accidents and traffic congestions caused by overheight vehicles attempting to enter locations that are too low for them. This application is compliant and relies on the US (WAVE) standards.

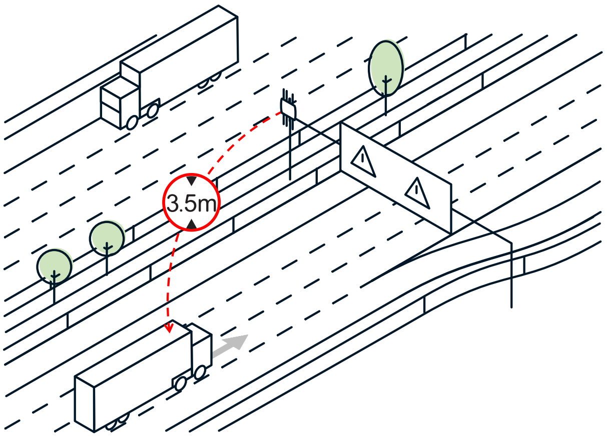

Figure 17. OHV Safety Application

5.3.2. TIW - OHV mode of operation

Model

- The OHV is a V2I application use case. that works for the **TrafficInfo** remote entity type
- The application uses object-in-area algorithms for filtering.
- The triggering conditions of the application are based on the [ITIS codes] and the configured speeding threshold.

Filter setup

The TIW application subscribes to Road Information, where:

- The host vehicle must be in the [relevance zone] of the **TrafficInfo** entity type. The relevance zone can be either a designated track or a circular area.

- The host vehicle must also be at the [same altitude] as the [relevance zone] of the TrafficInfo entity type.
- The heading of the host vehicle must be [aligned with the zone's direction]. In the WWR use case the heading is the "wrong way" relative to the direction of the zone.

Triggering conditions

- The application is triggered based on ITIS codes
- If the all of the following conditions are true, a Overheight Vehicle Warning use case will be assumed (and no specific notification is sent):
  - The following 3-code ITIS code sequence is found anywhere in the Road Information:
    - "Height Limit"
    - "0"−"255"
    - "Meters" or "Feet"
  - The configured height of the host vehicle is larger than the height limit minus **overheightThr**.
- If the TIW is triggered by a filtered object with Road Information, but none of the use cases match any triggering criteria, then a general TIW notification is triggered.

5.3.3. TIW and OHV configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.tiw: true**
- The OHV use-case has the following configuration options under the **ohv** key
  - **overheightThr**: Additional clearance that is subtracted from the actual height limit for the height check.
- The generic TIW application has no additional configuration.

5.3.4. OHV and general TIW notifications

The TIW application displays different notification levels based on the [TrafficInfo] type and the speeding threshold. In the WWR use case, the notification level is always ALERT. The generic TIW different notification levels are the following:

- [Alert] - this notification is displayed when the host vehicle is speeding within a [Regulation] or [Danger] type
- [Warning] - this notification is displayed when the host vehicle is speeding within any other type, for example [Advice]
- [Info] - this is an informative notification, displayed when the host vehicle is not speeding

The notification contains the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) and the Object Data and Relative Data of the remote vehicle.

In the OHV use case, the level is ALERT, if the vehicle is larger than the height limit, and WARNING otherwise (when it is within **overheightThr** of the actual height limit).

General TIW notifications also contain the use case, if multiple cases are triggered then the latest one.

TIW notifications also contain the ITIS code list of the Road Information object.

5.4. Road Event Warning (REW) - Hazardous Location Warning (HLW) use case

The Hazardous Location Warning (HLW) use case advises the driver of the host vehicle about a hazardous location ahead in the path of travel even if it is obscured from vision.

5.4.1. REW - HLW safety application overview

The Hazardous Location Warning (HLW) is a use case for the Road Event Warning (REW) safety application, which is an I2V communication based safety feature that informs the driver of the host vehicle about a temporary or permanent hazardous location (such as a fallen tree on the road ahead). This information is based on the received regional roadside event codes broadcast by the RSUs nearby.

The HLW helps drivers in avoiding unseen road hazards, increasing travel safety, and reducing the number of potential accidents. This application is compliant and relies on to the European (ETSI G5) standards.

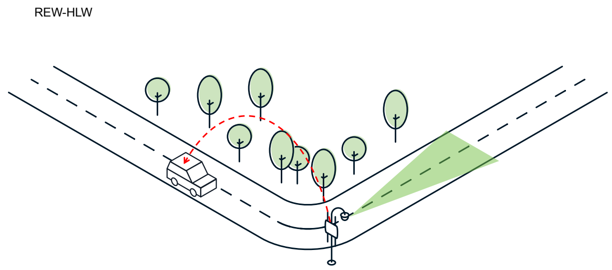

Figure 18. HLW Safety Application

5.4.2. HLW mode of operation

Model

- REW is an I2V application that uses dynamic road event information received from other cars and infrastructure devices. Road Events are dynamic, and may appear, move, or resolved within seconds. For static information handling, refer to the Traffic Information Warning safety application use cases.
- REW uses [lane abstraction] for the target area.

Filter setup

The application subscribes to **Road Information**, where the host vehicle is traveling in the relevant direction and in the awareness, detection, or relevance zones of the [Road Information object].

Triggering conditions

The REW application handles the following calculations on every processing cycle:

- If the HLW use case is enabled, and the type of the road information is any of the following, the use case will be set to Hazardous Location:
  - Hazardous Location: Surface Condition
  - Hazardous Location: Obstacle on the road
  - Hazardous Location: Animal on the road
  - Hazardous Location: Dangerous Curve
- If none of the REW use cases match or none of them are enabled, then the use case will be set to General REW.
- A REW notification is sent for every filtered object.

5.4.3. REW - HLW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The REW application is enabled if the **enable.rew** option or any of its use cases are enabled such as **enable.hlw: true**
- The application itself has the following configuration options under the key **rew**:
  - **vOn** and **vOff**: parameters of the speed-dependent automatic enable/disable.

5.4.4. REW notifications

REW is always an ALERT level notification.

The notification contains the use case of the warning (General, HLW, RWW, AWW, or TJAW), and the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) .

The notification also contains the zone in which the host vehicle is in (Detection, Awareness, or Relevance)

The REW notification contains the following data about the Road Event:

- The Object Data corresponding to the reference point of the event (for more information, see Common notification contents).
- The expiration of the Road Event as a Unix timestamp in ms.
- The type of the event.
- The subtype of the event.

5.5. Road Event Warning (REW) - Roadworks Warning (RWW) use case

The Roadworks Warning (RWW) use case informs the driver about ongoing roadworks in the path of travel ahead.

5.5.1. RWW safety application overview

The Roadworks Warning (RWW) is a use case of the Road Event Warning (REW) safety application, which is an I2V communication based safety feature that issues a warning to the driver of the host vehicle about roadworks in progress ahead in traffic in the same lane and direction of travel. This information is received from the nearby RSUs.

The RWW helps drivers to avoid accidents related to roadwork lane closures in the forward path of travel. This application is compliant and relies on to the European (ETSI G5) standards.

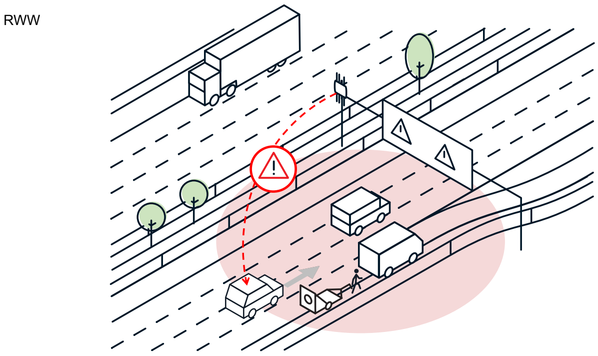

Figure 19. RWW Safety Application

5.5.2. RWW mode of operation

Model	• REW is an I2V application that uses dynamic road event information received from other cars and infrastructure devices. Road Events are dynamic, and may appear, move, or resolved within seconds. For static information handling, refer to the Traffic Information Warning safety application use cases. • REW uses [lane abstraction] for the target area.
Filter setup	The application subscribes to **Road Information**, where the host vehicle is traveling in the relevant direction and in the awareness, detection or relevance zones of the [Road Information object.]
Triggering conditions	The REW application handles the following calculations on every processing cycle: • If the RWW use case is enabled, and the type of the road information is Road Works, the use case will be set to Road Works. • If none of the REW use cases match or none of them are enabled, then the use case will be set to General REW.

- A REW notification is sent for every filtered object.

5.5.3. REW - RWW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The REW application is enabled if the **enable.rew** option or any of its use cases are enabled such as **enable.rww: true**
- The application itself has the following configuration options under the key **rew**:
  - **vOn** and **vOff**: parameters of the speed-dependent automatic enable/disable.

5.5.4. REW notifications

REW is always an ALERT level notification.

The notification contains the use case of the warning (General, HLW, RWW, AWW, or TJAW), and the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) .

The notification also contains the zone in which the host vehicle is in (Detection, Awareness, or Relevance)

The REW notification contains the following data about the Road Event:

- The Object Data corresponding to the reference point of the event (for more information, see Common notification contents).
- The expiration of the Road Event as a Unix timestamp in ms.
- The type of the event.
- The subtype of the event.

5.6. Road Event Warning (REW) - Adverse Weather Warning (AWW) use case

The Adverse Weather Warning (AWW) use case issues a warning about severe weather conditions in a target area.

5.6.1. REW - AWW safety application overview

The Adverse Weather Warning (AWW) is a use case for the Road Event Warning (REW) safety application, which is an I2V communication based safety feature that issues a warning to the driver of the host vehicle about severe weather conditions (such as strong winds, floods, or slippery roads due to heavy rain) in the direction of travel.

The AWW helps to inform drivers about hazardous road conditions caused by extreme weather. This application is compliant and relies on the European (ETSI G5) standards.

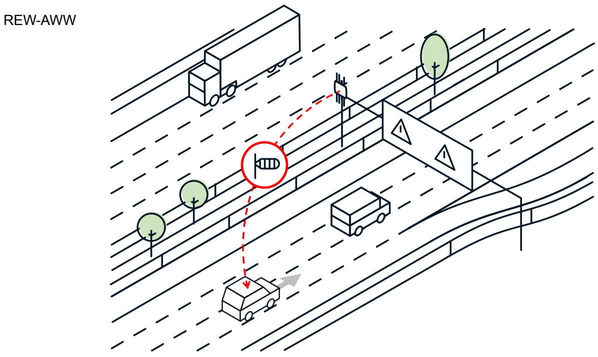

Figure 20. AWW Safety Application

5.6.2. HLW mode of operation

Model

- REW is an I2V application that uses dynamic road event information received from other cars and infrastructure devices. Road Events are dynamic, and may appear, move, or resolved within seconds. For static information handling, refer to the Traffic Information Warning safety application use cases.
- REW uses [lane abstraction] for the target area.

Filter setup

The application subscribes to **Road Information**, where the host vehicle is traveling in the relevant direction and in the awareness, detection, or relevance zones of the [Road Information object.]

Triggering conditions

The REW application handles the following calculations on every processing cycle:

- If the AWW use case is enabled, and the type of the road information is any of the following, the use case will be set to Adverse Weather:
  - Adverse Weather Condition: Adhesion
  - Adverse Weather Condition: Extreme Weather
  - Adverse Weather Condition: Visibility
  - Adverse Weather Condition: Precipitation
- If none of the REW use cases match or none of them are enabled, then the use case will be set to General REW.
- A REW notification is sent for every filtered object.

5.6.3. REW - AWW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The REW application is enabled if the **enable.rew** option or any of its use cases are enabled such as **enable.aww: true**
- The application itself has the following configuration options under the key **rew**:
  - **vOn** and **vOff**: parameters of the speed-dependent automatic enable/disable.

5.6.4. REW notifications

REW is always an ALERT level notification.

The notification contains the use case of the warning (General, HLW, RWW, AWW, or TJAW), and the Object Data of the host vehicle(for more information, see “Common notification contents” (p. 2)) .

The notification also contains the zone in which the host vehicle is in (Detection, Awareness, or Relevance)

The REW notification contains the following data about the Road Event:

- The Object Data corresponding to the reference point of the event.
- The expiration of the Road Event as a Unix timestamp in ms.
- The type of the event.
- The subtype of the event.

5.7. Road Event Warning (REW) - Traffic Jam Ahead Warning (TJAW) use case

The TJAW use case issues a warning about severe weather conditions in a target area.

5.7.1. REW - TJAW safety application overview

The Traffic Jam Ahead Warning (TJAW) is a use case for the Road Event Warning safety application, which is an I2V communication based safety feature that issues a notification to the driver of the host vehicle about Traffic Conditions in the direction of travel.

The TJAW helps to inform drivers about traffic congestions, their state and severity, so they can make informed decisions about choosing the best route. This application is compliant and relies on the European (ETSI G5) standards.

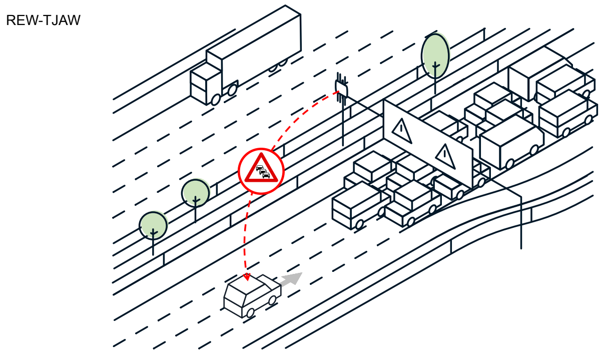

Figure 21. TJAW Safety Application

5.7.2. HLW mode of operation

Model

- REW is an I2V application that uses dynamic road event information received from other cars and infrastructure devices. Road Events are dynamic, and may appear, move, or resolved within seconds. For static information handling, refer to the Traffic Information Warning safety application use cases.
- REW uses [lane abstraction] for the target area.

Filter setup

The application subscribes to **Road Information**, where the host vehicle is traveling in the relevant direction and in the awareness, detection or relevance zones of the [Road Information object].

Triggering conditions

The REW application handles the following calculations on every processing cycle:

- If the TJA use case is enabled, and the type of the road information is Traffic Condition, and the subtype is any of the following 1, the use case will be set to Traffic Jam Ahead:
  - Traffic jam slowly increasing
  - Traffic jam strongly increasing
  - Traffic jam stationary
  - Traffic jam slightly decreasing
  - Traffic jam decreasing
  - Traffic jam strongly decreasing
- If none of the REW use cases match or none of them are enabled, then the use case will be set to General REW.
- A REW notification is sent for every filtered object.

5.7.3. REW - TJAW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The REW application is enabled if the **enable.rew** option or any of its use cases are enabled such as **enable.tjaw: true**
- The application itself has the following configuration options under the key **rew**:
  - **vOn** and **vOff**: parameters of the speed-dependent automatic enable/disable.

5.7.4. REW notifications

REW is always an ALERT level notification.

The notification contains the use case of the warning (General, HLW, RWW, AWW, or TJAW), and the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) .

The notification also contains the zone in which the host vehicle is in (Detection, Awareness, or Relevance)

The REW notification contains the following data about the Road Event:

- The Object Data corresponding to the reference point of the event.
- The expiration of the Road Event as a Unix timestamp in ms.
- The type of the event.
- The subtype of the event.

5.8. Traffic Signal Information (TSI) - Time-To-Green (TTG) use case

The Time-To-Green (TTG) use case advises the driver of the host vehicle about the time remaining until the next green light signal phase of the traffic light in the lane ahead.

5.8.1. TTG safety application overview

Time-To-Green (TTG) is a use case of the Traffic Signal Information (TSI) safety application, which is a I2V communication based safety feature that provides information about the time remaining until the next green signal interval of the traffic light ahead within the same lane. This information is based on the received Traffic Signal Information.

TTG helps drivers in avoiding or mitigating delays as a result of stop-and-go motions through intersections, optimizing fuel efficiency, and reducing traffic congestion. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

TTG

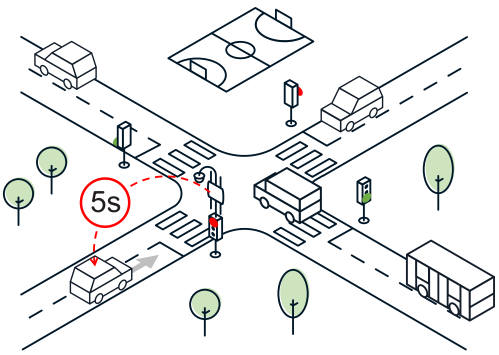

Figure 22. TTG Safety Application

5.8.2. TTG mode of operation

Model	• TSI is an I2V application that uses traffic signal information received from intersections. All TSI use cases work best when future traffic signal phase information is also known. • TSI uses [lane abstraction] for the target area.
Filter setup	The application subscribes to **Intersection** Information, where the host vehicle is in an ingress lane and it is not traveling in the wrong direction.
Triggering conditions	The TSI application handles the following calculations on every processing cycle: • The following data is calculated: • The distance to be traveled to the stop line of the ingress lane. • The timestamp when the stop line will be reached, assuming constant velocity.

- If the **useTurnSignal** option is on, connections that are not predicted to be taken based on the turn signal status of the host vehicle are skipped.
- For each remaining connection, if the current phase is not green, and the start time of the next green phase is available, the connection is added to the TTG list.
- If the TTG use case is enabled, and the TTG list is not empty, a TTG notification is sent.
- If both **likelyTime** and **minEndTime** fields are available for a phase, the **likelyTime** field is used if the **preferLikelyTime** option is enabled, the **minEndTime** otherwise. If only one of these is available, that field is used, regardless of the configuration.

5.8.3. TSI - TTG configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The TSI application is enabled if any of its use cases are enabled such as **enable.ttg: true**
- The following parameters can be configured in the **tsi** key:
  - **useTurnSignal**: Predict the lane connection and use only that connection for notifications if turn signal status is available. If this option is disabled, all lane connections are considered.
  - **preferLikelyTime**: Use the **likelyTime** field of the traffic signal information instead of the **minEndTime** for phase end time calculation, if both are present.

5.8.4. TTG notifications

TTG is always an INFO level notification.

The notification contains the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) and the Object Data and Relative Data of the remote vehicle.

The notification also contains the lane ID of the ingress lane as well as the distance to the stop line of the ingress lane.

The TTG notification contains the speed advices with the signal group IDs and the time to green in ms.

5.9. Traffic Signal Information (TSI) - Red Light Violation (RLV) use case

This safety application advises the driver of the host vehicle about the time remaining until the next green light signal phase of the traffic light in the lane ahead.

5.9.1. TSI - RLV safety application overview

Red Light Violation (RLV) is a use case of the Traffic Signal Information (TSI) safety application, which is an I2V communication based safety feature that provides a warning when the vehicle is about to run a red light at an intersection. This information is based on the received Traffic Signal Information.

The RLV helps drivers in avoiding hazards at intersections that result from running a red light and reducing traffic congestion. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

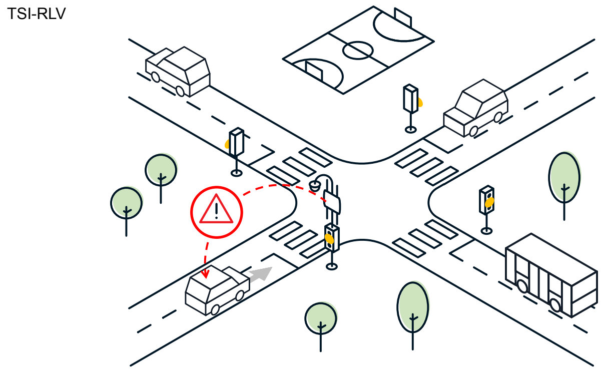

Figure 23. RLV Safety Application

5.9.2. RLV mode of operation

Model

- TSI is an I2V application that uses traffic signal information received from intersections. All TSI use cases work best when future traffic signal phase information is also known.
- TSI uses [lane abstraction] for the target area.

Filter setup

The application subscribes to **Intersection** Information, where the host vehicle is in an ingress lane and it is not traveling in the wrong direction.

Triggering conditions

The TSI application handles the following calculations on every processing cycle:

- The following data is calculated:
  - The distance to be traveled to the stop line of the ingress lane.
  - The timestamp when the stop line will be reached, assuming constant velocity.
- If the **useTurnSignal** option is on, connections that are not predicted to be taken based on the turn signal status of the host vehicle are skipped.
- If the current phase is red, and the end time is after the stop line timestamp, the connection is added to the RLV list.
- If the current phase is not red, the start and end times of the next red phase are available, and the stop line timestamp is within this interval, the connection is added to the RLV list.
- If the RLV use case is enabled, and the RLV list is not empty, a RLV notification is sent.
- If both **likelyTime** and **minEndTime** fields are available for a phase, the **likelyTime** field is used if the **preferLikelyTime** option is enabled, the **minEndTime** otherwise. If only one of these is available, that field is used, regardless of the configuration.

5.9.3. TSI - RLV configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The TSI application is enabled if any of its use cases are enabled such as **enable.rlv: true**
- The following parameters can be configured in the **tsi** key:
  - **useTurnSignal**: Predict the lane connection and use only that connection for notifications if turn signal status is available. Disabling this option will send a notification when a VRU is detected on a possibly irrelevant lane connection.
  - **preferLikelyTime**: Use the **likelyTime** field of the traffic signal information instead of the **minEndTime** for phase end time calculation, if both are present.

5.9.4. RLV notifications

RLV is always an ALERT level notification.

The notification contains the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) and the Object Data and Relative Data of the remote vehicle.

The notification also contains the lane ID of the ingress lane as well as the distance to the stop line of the ingress lane.

The RLV notification contains the red light information with the signal group IDs and the start and end timestamps of the red phase.

5.10. Traffic Signal Information (TSI) - Green Light Optimized Speed Advise (GLOSA) use case

This safety application advises the driver of the host vehicle about the recommended speed that will result in the vehicle arriving at the next intersection on a green light signal.

5.10.1. GLOSA safety application overview

The Green Light Optimized Speed Advise (GLOSA) is a use case for the Traffic Signal Information (TSI) safety application, which is an I2V communication based safety feature that suggests a calculated speed to the driver of the host vehicle, allowing them to pass through an intersection during the green signal interval. This information is calculated based on the received Traffic Signal Information and the current speed and location of the host vehicle.

The GLOSA helps drivers in avoiding or mitigating delays as a result of stop-and-go movements through intersections, optimizing fuel efficiency, and reducing traffic congestion. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

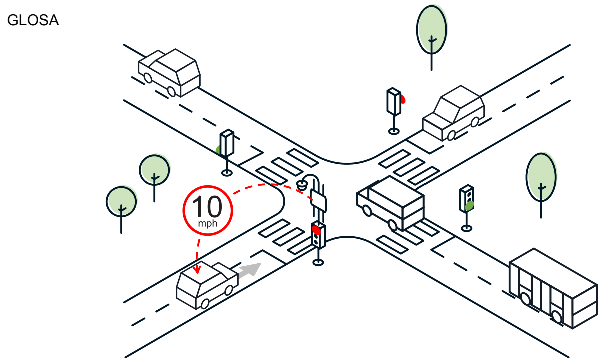

Figure 24. GLOSA Safety Application

5.10.2. GLOSA mode of operation

Model

- TSI is an I2V application that uses traffic signal information received from intersections. All TSI use cases work best when future traffic signal phase information is also known.
- TSI uses [lane abstraction] for the target area.

Filter setup

The application subscribes to **Intersection** Information, where the host vehicle is in an ingress lane and it is not traveling in the wrong direction.

Triggering conditions

The TSI application handles the following calculations on every processing cycle:

- The following data is calculated:
  - The distance to be traveled to the stop line of the ingress lane.
  - The timestamp when the stop line will be reached, assuming constant velocity.

- If the **useTurnSignal** option is on, connections that are not predicted to be taken based on the turn signal status of the host vehicle are skipped.
- If the vehicle is traveling too slow or too fast compared to the advised speeds (see below), the connection is added to the GLOSA list.
- If the GLOSA use case is enabled, and the GLOSA list is not empty, a GLOSA notification is sent.
- If both **likelyTime** and **minEndTime** fields are available for a phase, the **likelyTime** field is used if the **preferLikelyTime** option is enabled, the **minEndTime** otherwise. If only one of these is available, that field is used, regardless of the configuration.
- The actual GLOSA value for each connection is calculated as follows:
  - Based on the start and stop times of the current or next green light and the distance to the stop line, a min and max advised speed is calculated.
  - If the vehicle is traveling too fast, the advice is set to the calculated minimum speed. If this speed is less than 20 km/h, it is set to zero.
  - If the vehicle is traveling too slow, the advice is set to the calculated maximum speed. If this speed is more than any of the following, the advice is set to zero.
    - Speed limit of the lane
    - 1.5 times the current speed
    - 70 km/h

5.10.3. TSI - GLOSA configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The TSI application is enabled if any of its use cases are enabled such as **enable.glosa: true**
- The following parameters can be configured in the **tsi** key:
  - **useTurnSignal**: Predict the lane connection and use only that connection for notifications if turn signal status is available. If this option is disabled, all lane connections are considered.
  - **preferLikelyTime**: Use the **likelyTime** field of the traffic signal information instead of the **minEndTime** for phase end time calculation, if both are present.

5.10.4. GLOSA notifications

TTG is always an INFO level notification.

The notification contains the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) and the Object Data and Relative Data of the remote vehicle.

The notification also contains the lane ID of the ingress lane as well as the distance to the stop line of the ingress lane.

The TTG notification contains the speed advices with the signal group IDs and the speed advice itself.

5.11. Vulnerable Road User (VRU)

This safety application warns the driver of the host vehicle about driving into a lane with a currently occupied pedestrian crossing.

5.11.1. VRU safety application overview

The Vulnerable Road User (VRU) safety application is an I2V communication based safety feature that issues a warning to the driver if their predicted path intersects with a pedestrian crossing that is currently marked as occupied - even if the pedestrian crossing is obstructed from view. This application is triggered based on the lane information of the surrounding area and the turn indication data of the host vehicle. The current occupancy of the nearby pedestrian crossings comes from the various sensor information flags of the intersection (for example camera or motion sensor data related to the pedestrian crossing).

The VRU safety app helps drivers to avoid or mitigate the risk of accidents in pedestrian crossings, increasing safety, and reducing traffic congestion. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

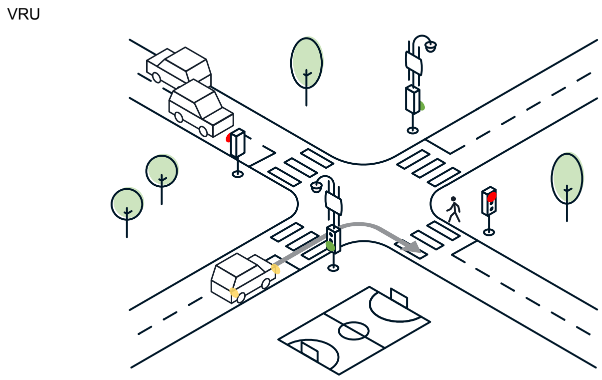

Figure 25. VRU Safety Application

5.11.2. VRU mode of operation

Model	• The application is enabled or disabled based on the speed of the host vehicle (for more information, see “Speed-dependent behavior” (p. 3)). • VRU uses [lane abstraction] for the target area.
Filter setup	The application subscribes to **Intersection** Information, where the host vehicle is in an ingress lane and it is not traveling in the wrong direction.
Triggering conditions	The application handles the following calculations on every processing cycle: • The current lane of the host vehicle is selected • If the distance of the host vehicle from the end of the lane (the stop line) is more than checkDist, processing stops for this intersection • A lane connection will be included in the notification, if all of the following criteria are met: • The connection starts at the current lane. • The connection is marked dangerous. • The vehicle is predicted to take the connection based on its turn signals, the turn signal status of the host vehicle is not available, or the **useTurnSignal** configuration is not enabled. • If any connections match, a notification is sent.

5.11.3. VRU configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.vru: true**
- The following parameters can be configured in the **vru** key
  - **checkDist**: The maximum distance to the stop line of the current lane. If the host vehicle is further than this, no notifications are sent
  - **useTurnSignal**: Predict the lane connection and use only that connection for notifications if turn signal status is available. Disabling this option will send a notification when a VRU is detected on a possibly irrelevant lane connection.
  - **predictionHorizon**: Experimental feature. Do not use!

- **method**: Experimental feature, do not set to other than ConnectionState.

5.11.4. VRU notifications

The application sends ALERT level notifications if any of the following criteria are met:

- The vehicle is predicted to take the connection based on its turn signal.
- The turn signal status of the host vehicle is not available.

A WARNING level notification is sent in any other case.

The notification contains the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) and the Object Data and Relative Data of the remote vehicle.

The notification also contains a list of dangerous egress lanes, with their lane IDs and the notification level of the lane according the rules mentioned above.

5.12. Stationary Vehicle Warning (SVW) - Infrastructure-augmented (ISV) use case

The Stationary Vehicle Warning (SVW) safety application, used in Infrastructure-augmented (ISV) mode sends a notification to the driver when there is possibly unseen a stationary vehicle in the predicted egressing lane of an intersection.

5.12.1. SVW safety application overview

The Stationary Vehicle Warning (SVW) safety application is a V2V communication based safety feature that issues a warning to the driver of the host vehicle in case of a remote vehicle is stationary ahead in traffic in the same lane and direction of travel—even if the remote vehicle is obstructed from view.

The SVW helps drivers in avoiding or mitigating front-to-rear vehicle collision in the forward path of travel with a stationary vehicle. This application is compliant and relies on to the European (ETSI G5) and the US (WAVE) standards.

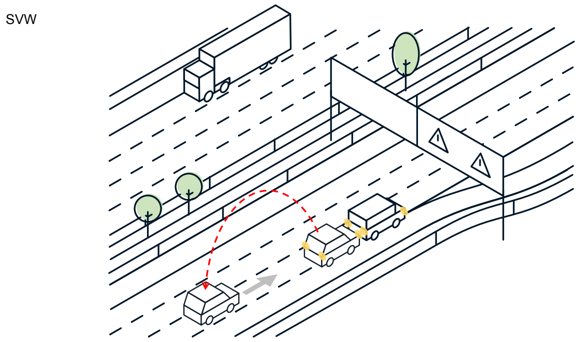

Figure 26. SVW Safety Application

5.12.2. SVW mode of operation

Model	• SVW is an application that works for the **Vehicle** remote entity type. • The application is enabled or disabled based on the speed of the host vehicle (for more information, see “Speed-dependent behavior” (p. 3)). • SVW uses [lane abstraction] for the target area.
Filter setup	The application subscribes to object-intersection relations, meaning all [Object type entities] that are in a lane of an intersection and all of the related [Intersection type entities].
Triggering conditions	The application handles the following calculations on every processing cycle:

- Detects which lane the host vehicle is located in. If the host vehicle is not in an intersection, not in an ingress lane, or is going the wrong way, the processing stops.
- Collects all remote objects that match the following criteria:
  - Stationary (its speed is less than **stationarySpeedThr**).
  - It is closer to the stop line (taking the remote vehicle entity's length into account, if available) than **stopLineDistThr** .
- Predicts which egress lanes are possible destinations, based on the connections of the ingress lane at the intersection. An egress lane is considered as a possible destination, if the host vehicle is predicted to take the connection based on its turn signals, the turn signal status of the host vehicle is not available, or the **useTurnSignal** configuration is not enabled.
- For each possible egress lane, if there is a stationary remote object in the lane, a notification is sent.

5.12.3. SVW configuration parameters

The following parameters must be set In the configuration space of **saf.json**:

- The application is enabled by **enable.svw: true AND svw.triggerForAbnormalVehicles: true**
- The following parameters can be configured in the **svwW** key
  - ISV use case specific settings can be configured under the **svw** key
    - **targetLength**: The base length of the target area [m]
    - **useAdaptiveTarget**: Adapt effective **targetLength** in proportion to the host vehicle's speed
    - **ttc**: Time-to-collision threshold
    - **useAdaptiveTtc**: Use speed-dependent TTC scaling. It is recommended to set either this or **useAccelerationForTtc** to true.
    - **useAccelerationForTtc**: Assume constant acceleration when calculating the minimum distance point. It is recommended to set either this or **useAdaptiveTtc** to true.
    - **stationarySpeedThr**: The speed threshold, below which a remote object is considered stationary.

5.12.4. SVW notifications

The application sends ALERT level notifications.

The notification contains the Object Data of the host vehicle (for more information, see “Common notification contents” (p. 2)) and the Object Data and Relative Data of the remote vehicle.