AI CSB vehicle Enhancements (ACE)

1.5.3

December 2019

Introduction

Purpose

This modification primarily revolves around enhancing the way in which AI-controlled instances of CSB vehicles behave. More specifically it seeks to enable the AI to:

- Leverage more vehicle functions that it is by default – i.e., under standard M+R-derived scripts – capable, such as heating and cooling.
- *Consistently* vary its preferences and consequent behavior – i.e., what functions to employ under what circumstances – on a per-vehicle and/or per-context basis, such as one vehicle "choosing" to activate its lights earlier than another. Consistency is significant: In spite of choices and actions undertaken by AI vehicles being to some extent subjective, they still ought to consistently reflect the same set of rules and principles, such that actions undertaken by one vehicle at present time do not contradict past actions undertaken by the same vehicle. For example, a vehicle may choose to open some or all of the available folding passenger windows when the temperature of the environment is perceived as "neutral". While both the temperature perception and the choice of which windows to open is subjective and variable, it must hold that under "warm" conditions at least the same or more windows than under "neutral" conditions are to be opened by that same vehicle; behaving the other way around, and/or basing window state on pure chance, while certainly easier to implement, would lack consistency.
- Adjust vehicle state non-instantaneously, so as to – to some extent at least – appear more human-like; for instance by not activating the wipers at the very instant of the very first rain droplet falling down from the sky, but after a variable delay. In other words, AI vehicles shall exhibit traits of *tolerance* and *inertia* ("laziness"), like us humans do. Furthermore, at the degree practically feasible, AI vehicle state adjustment is to occur via adjustment of the very same cockpit controls available to users – not in some magical, *Deus ex machina* kind of way.

Nonetheless, along the way of implementing such AI-dedicated changes, a number of smaller user-level improvements to apposite subsystems made it into this modification as well.

The purpose of this document is, predominantly, to provide a brief, non-technical overview of the most significant among both kinds of changes. For the sake of readability, the language used to describe individual changes is intentionally somewhat abstract – the intent is to paint the general, vehicle-neutral idea behind each major change / feature, while glossing over details that are specific to individual vehicles, models, or variants. The reader is encouraged to refer to the scripts and in-line commentary therein when greater accuracy on implementation and integration details, and/or rationale behind implementation choices is sought.

Supported CSB vehicles

Currently support is limited to the solo Euro 3 variants ("ETP 3" and "ETP 11") of the (first-generation) MB O530 CSB module.

Compatibility

The current version of this modification was developed in conjunction with, and tested against, version 1.40 of the CSB modification package, and OMSI version 2.3.004.

General project scope and direction

Even though as of this release several solid months of effort have already been invested into this project, it is by no means perfect or finished; at least as far as the central theme of simulating human behavior is concerned, sky is the limit. That being said, it should also be stressed that this project does by no means intend to serve as a "silver bullet" to every OMSI-AI-related shortcoming and frustration there is. All it strives to do is more extensively leverage existing tools, and in particular the scripting engine, without otherwise resorting to exotic workarounds. Therefore, inherent limitations of the game – such as AI vehicles being unable to dynamically vary their paths – are and always will be the "hard upper bound" of this project's scope.

That aside, future versions of this modification may bring support to other vehicles of the CSB "family" as well.

Whether and when future versions materialize is a function of time availability on the developers' part and community-received feedback.

Feedback

Users are welcome and encouraged to submit constructive feedback on this modification, via the overarching CSB modification's OOF discussion thread.

Credits

The following individuals have had direct involvement in this endeavor:

Unorthodox Paradox was the primary implementation and documentation author.

citaro142, coordinator of the "upstream" CSB modification, contributed the functional requirements of the AI lighting module, and a wealth of real-world observations. He also assisted with beta-testing.

Florito contributed insight on technical aspects of simulated components, and inspiration on future work. He also assisted with beta-testing and debugging.

The following individuals are acknowledged for having provided the foundation directly underlying this work:

Morphi has been tirelessly refining the Citaro's script base from its release to this day. His many contributions include significant VDV and transmission enhancements.

alTerr, *wizard*, et al., being the team behind the original OMSI Citaro, carried out the groundwork of adapting the M+R MAN script base to the Citaro's needs.

Marcel Kuhnt, *Rüdiger Hülsmann*, et al., authored the original MAN script base, still heavily utilized by virtually all OMSI buses at present.

Terms of use

This modification is provided “as-is”, without any warranty.

Original or substantially altered content (such as completely new scripts, as well as significant patches to preexisting ones), distributed as part of this modification, is subject to the terms of the CC0 1.0.

Other redistributed content (such as slightly modified 3D objects and textures, as well as trivially-patched scripts) is subject to the terms of its respective owners.

Index

1. Installation and uninstallation

Installation

1. Ensure that the base CSB modification package, version 1.40, including its dependencies, is correctly installed.
2. Uninstall any earlier version of this modification already installed (see next section).
3. Extract the `Texture` and `Vehicles` directories of the downloaded compressed archive into your root OMSI 2 installation directory. When prompted whether directory merging is desired, respond "Yes"; no preexisting files will be modified.
4. From Morphi's MB O530 package, version 4.4 or 4.5, residing, by default, under `<OMSI2>\Vehicles\MB_O530_Modded\Sound-Citaro`, copy the following sound files to `<OMSI2>\Vehicles\CSB_J_MB_O530\Sound-Citaro\ACE\imported`:
   - `pedal-bremse.wav`
   - `pedal-gas.wav`
   - `pedal-kickdown.wav`
   - `reversewarn.wav`
5. Additionally from Morphi's MB O530 package, this time residing, by default, under `<OMSI2>\Vehicles\MB_O530_Modded\Model\O530`, copy the file named `wagennummer.o3d` to `<OMSI2>\Vehicles\CSB_J_MB_O530\Model\ACE\O530\imported`.
6. Optionally, if you intend to use the testing / diagnostic function of the AI lighting subsystem (discussed in a Appendix C), you may want to register with the game some additional cloud textures. To do so, copy the contents of the `append_to_clouds.cfg` file, to be found at the root of the downloaded compressed archive, and paste them into `<OMSI2>\Weather\clouds.cfg`.

A successful installation will enable in-game selection and AI-list referencing of the vehicles summarized below. Each of these vehicles is an AI-enhanced counterpart of the corresponding base CSB modification original.

Manufacturer	Model	Characteristics	.bus-file	Repaint directory
CSB - Euro Traffic Partner	ETP 3 - ACE	MB O530 / 2-door / E3 / ZF	`<OMSI2>\Vehicles\CSB_J_MB_O530\CSB_ETP_3_ACE.bus`	`<OMSI2>\Vehicles\CSB_A_Lackierungen\rep_CSB_ETP_3\ACE`
	ETP 11 - ACE		`<OMSI2>\Vehicles\CSB_J_MB_O530\CSB_ETP_11_ACE.bus`	`<OMSI2>\Vehicles\CSB_A_Lackierungen\rep_CSB_ETP_11\ACE`

Uninstallation

Search for, and delete, all files and directories that include the acronym “ACE” as part of their path name.

Most of the modification’s files already reside under dedicated sub-directories named “ACE”:

- <OMSI2>\Vehicles\
  - CSB_A_Lackierungen\
    - rep_CSB_ETP_3**ACE**
    - rep_CSB_ETP_11**ACE**
  - CSB_J_MB_O530\
    - Model**ACE**
    - Script**ACE**
    - Sound-Citaro**ACE**
    - Texture**ACE**

There are, nonetheless, some files that still share, for reasons of practicality, directories with the base modification, and must therefore be removed individually:

- <OMSI2>\Vehicles\CSB_J_MBO530\
  - CSB_J_MB_O530**CSB_ETP_3_ACE.bus**
  - CSB_J_MB_O530**CSB_ETP_11_ACE.bus**
  - Model\
    - **model_CSB_ETP_3_ACE.cfg**
    - **model_CSB_ETP_11_ACE.cfg**
    - **passengercabin_O530_ACE.cfg**
  - Sound-Citaro\
    - **sound_om906hla_E3_ZF_ACE.cfg**
    - **sound_om906hla_E3_ZF_ACE_unfocused.cfg**

2. Significant AI-specific changes

2.1. Exterior lighting

Different AI vehicles will employ different kinds of exterior light sources under different conditions and for different reasons.

Conditions are sets of a) solar elevation angle, b) environmental brightness, c) precipitation, d) season, and e) temperature ranges, subjectively interpreted by each vehicle. Depending on the interpretation, each vehicle is said to effectively be in either a) a *seeing*, or b) an *adverse-seeing*, or c) a *reduced-seeing*, or d) a *being-seen*, or e) an *off* context. Depending on the context, different light sources will be used, based on, once again, subjective preferences of each vehicle.

Seeing

A *seeing* context is when a vehicle deems that its environment is too dark to adequately make out unassisted; for example at night, or during the day, when the weather is overcast. Every vehicle will enter a seeing context at some point. Most vehicles will then switch on their headlights. Some will additionally switch on their head fog lights, if equipped with such.

Reduced-seeing

A *reduced-seeing* context is entered when the environment is only deemed borderline too dark; for example a little before sunrise, on a clear day. Most vehicles will not differentiate between seeing and reduced-seeing, and hence their preferred exterior light sources will remain unaffected. A minority of vehicles will however differentiate, as a consequence only activating their standstill lights under such conditions, potentially combined with head fog lights, if available.

Adverse-seeing

The *adverse-seeing* conditions are another subset of the baseline *seeing* ones. Here the environment is considered to be both dark *and risky* in some way – for example when visibility is impaired due to heavy rain at night. Most vehicles *will* differentiate between seeing and adverse-seeing. Those that do will employ head or full (head + tail) fog lights, in addition to headlights, while in this context.

Being-seen

Being-seen conditions start when *seeing* conditions (including their two subsets) end; the two are mutually-exclusive. Such a context is typically entered at times when the sun glare warrants usage of lights for signaling presence to other traffic participants, as opposed to illuminating the environment; e.g. after sunrise and until the sun has ascended above a certain subjective threshold, ceasing to impair the driver's vision. Most AI vehicles, however, particularly those equipped with DRL, will lack such safety-consciousness, and will instead transition to an *off* context once seeing conditions no longer apply. Vehicles that *do* have such sensitivities, on the other hand, will use either a) standstill lights, or

b) standstill and head fog lights, or c) headlights, or d) headlights and head fog lights under being-seen conditions.

Off

This context holds whenever neither *seeing* nor *being-seen* conditions apply. All vehicles will then unconditionally switch off any exterior lights previously in use.

Temporarily-off

This context is entered when vehicles are stopped at prolonged[1], typically terminal stops, where told by OMSI to turn the engine off. Depending on whether the engine is indeed deactivated, as addressed in a later section, two sub-cases can manifest:

- If the engine indeed gets turned off, most vehicles will be satisfied by the automatic reduction of headlights (if previously active) to standstill-only, and will opt not to undertake any further manual exterior lighting reduction. Some more environmentally-friendly simulated drivers will however manually completely deactivate all exterior lights, until departure time.
- Otherwise, that is, if the engine for some reason is left running, vehicles will unconditionally manually reduce exterior lights to standstill (or standstill + fog lights) until departure time. Of course some will still eagerly opt to disable lights altogether.

Closing notes

- For most vehicles, *seeing* conditions will become effective *before* the time OMSI instructs them to turn their lights on; hardly any vehicles will wait until it gets really dark before deciding lights are needed.
- For most vehicles, *being-seen* conditions, if at all applicable, end at some point before midday[2]. For some vehicles, nonetheless, being-seen conditions will *always* apply when seeing conditions do not. Such vehicles will as a consequence never completely disable their exterior lights (with the potential exception of entering a *temporarily-off* context).
- *Temporarily-off* conditions are more likely to manifest during the day, particularly when the baseline / overarching context is that of *being-seen*.
- Light sources preferred under one context affect those preferred under other contexts; for example, vehicles relying on head fog lights for *being-seen* are more likely to also rely on them for the purpose of *seeing*.
- Perception – the classification of conditions to the aforementioned contexts / labels – and associated light source usage preferences are *static*[3].

---

1 For the purposes of this document, a prolonged stop is either a) a trip's first, second, or last stop entry, or b) any other stop where OMSI instructs the vehicle to turn its engine off.

2 This does of course vary depending on the season, due to the sun remaining at lower angles most of the day throughout the winter.

3 *Static* AI vehicle attributes are those that yield the same outcome under the same primitive, objective conditions (e.g. temperature). They are initialized once, when the vehicle is spawned or otherwise first assigned to the AI, and remain constant thereafter, until either a) OMSI is terminated or a different map gets loaded, or b) the user takes over control of the vehicle and subsequently reassigns it to the AI (in which case a new AI vehicle/driver profile gets generated, since it might be a different virtual colleague then taking over). Static attributes are easier to implement and yield better performance, at the cost of reduced realism.

- Perceptual and adjustment "latency" / tolerance – how much time will elapse between a) primitive conditions changing, and b) the vehicle acknowledging the fact that the context changed, and ultimately c) the vehicle reacting to the contextual change by adjusting the state of its lights – is *dynamic*[3]. Generally vehicles will be *more tolerant* of "good" conditions and *less tolerant* of "bad" conditions, that is, they will tend to be *faster* at responding to deteriorating conditions, and *slower* at responding to improving conditions.

2.2. Interior lighting

Activation and deactivation

Most vehicles will only decide to switch their interior lights on when the environment starts becoming *visibly* darker, as opposed to[4] exterior lights, which may be activated well before that time. The conditions for activation of interior lighting are in most cases a "worse" subset of the exterior lighting *seeing* conditions discussed previously. Vehicles that always use exterior lights for *being-seen*, i.e., vehicles that never deactivate their exterior lighting, no matter how favorable the environment, pose a notable exception, however: Some of them will never deactivate their interior lighting either[5].

Dimmed vs full mode

Typical city buses offer two interior lighting modes: A full and a dimmed one, with the former usually translating to all interior light sources being activated, and the latter translating to just some sources (usually excluding those directly behind the driver cubicle) being activated.

Under conditions of low brightness[6] most vehicles will employ the dimmed mode in order to reduce reflection glare on the windshield impairing their vision. Such vehicles will only transition their interior lighting to full mode at times when they would normally disable interior lights, were the weather conditions better; e.g., on cloudy days. As always there are exceptional minorities: Some vehicles will always favor the dimmed mode, and even fewer will always favor the full mode.

Dynamic attributes on the other hand are those that may yield a variety of outcomes under the same primitive objective conditions. Inversely to static attributes, they require greater implementation effort and yield worse performance, but benefit realism.

The question of which approach to favor is dealt with on a case-by-case basis. The general criterion relied on is whether the particular AI perceptual or behavioral aspect only occurs infrequently, and is thus less susceptible to being scrutinized as overly predictable or "mechanical" by the observing user. If the answer is positive, then static attributes are used; otherwise dynamic attributes.

4 Because most real-world drivers appear to value road safety more than passenger comfort.

5 Because an actual driver choosing to unconditionally leave exterior lighting on during the day is somewhat more likely to forget to disable interior lighting on time.

6 Which means "most of the time for most vehicles", given that the majority of vehicles will not use interior lights unless the environment is to some extent dark.

Temporary deactivation on service trips

A fair percentage of vehicles will opt for disabling interior lighting when undergoing a service trip. Most such vehicles will only do so under really dark conditions though, as well as early[7] in the morning and late in the evening.

Temporary deactivation at prolonged stops

A very small vehicle minority will disable[8] interior lights at stops when the engine is to be turned off, according to OMSI, as long as no passengers remain on board.

Closing notes

- For most vehicles, the activation conditions for interior lights will become effective at *roughly the same* time OMSI instructs them to turn them on. In contrast to exterior lighting seeing conditions, however, it is not entirely uncommon to also witness vehicles choosing not to activate interior lights before *actual sunset*.
- As with exterior lighting, perception and preferences are static, while adjustment delays are dynamic. Furthermore, adjustment delays are longer when interior lights are to be switched off[9] than when they are to be switched on.

2.3. Doors

Opening

Doors can be opened *lazily* or *eagerly*.

Lazy opening of a particular door occurs when at least one AI passenger requests entry or exit via that door.

7 Early vs. late determined on the basis of actual 24-hour time, not solar elevation angle. During those periods, on most maps, service trips will tend to last longer and be more frequent.

8 Perhaps for the driver to take a nap or simply enjoy some relaxing, undisturbed alone-time; or maybe to just conserve battery load.

9 Exception: *Dimming* (full → dimmed) occurs faster than *un-dimming* (dimmed → full), due to the glare irritating the driver.

Eager opening of one or multiple doors occurs when the vehicle a) is at a prolonged stop and OMSI instructs it to turn its engine off, and b) deems the environment as appropriate[10, 11] for door opening.

Closing

Doors can likewise be closed *lazily* or *eagerly*.

Lazy closing of all doors occurs when it is time to depart.

Eager closing of one or multiple doors occurs when either a) the environment is no longer perceived as appropriate[12] for those doors to be open, or b[13]) the current stop is non-prolonged and the doors appear not to be (anymore) in use by passengers boarding or disembarking.

Open front door departure

Rarely, a certain minority[14] of vehicles may depart from a stop without[15] closing the front door. This is most likely to happen when:

- The environmental temperature is borderline uncomfortably warm, yet not warm enough to use the A/C – according to the vehicle's perception.
- The weather is perceived as dry and clear, and it is daytime.
- The vehicle is undergoing a service trip or parking maneuver (see next section), or is at the last (pre-terminus) or first few stops of its trip.

---

10 Environmental perception influencing door and door wing locking, heating / cooling, window and hatch, and at some degree sun blind and wiper state, is driven as follows:
- The weather temperature is subjectively interpreted as being *coldest*, *cold*, *chilly*, *ideal*, *warm*, *hot*, or *hottest*.
- The solar elevation angle is subjectively interpreted as *day* or *night*.
- The precipitation rate, if any, is subjectively interpreted as *dry* or *wet*.

The interpretation is static, while the acknowledgment delay of perceptual change is dynamic. Conditions that are "worse than before" tend to be registered faster than conditions that are "better than before". For the algorithm of what constitutes a "better" vs. a "worse" transition, refer to the implementation of `ai_pre_cockpit.osc:ai_pre_cockpit_env`.

11 In general the likelihood of eager door opening occurring is proportional to temperature, daytime, and dryness. There is however an implicit personality trait involved as well, resulting in some vehicles being overall more inclined to unconditionally open their doors than others.

12 Note that environmental "inappropriateness" here does not necessarily translate to the complement of "appropriateness"; the distinction is asymmetrical. A vehicle may, upon *lazy* opening, and under certain conditions that are neither deemed as "really warm" nor "really cold", choose the middle ground of "lazily-doing-nothing", i.e., leaving the door open, even though it wouldn't have opened it in the first place if no boarding or disembarking passengers had requested its opening.

13 This allows vehicles to appear to close doors "asynchronously", as if in a hurry to depart, when no longer needed open, prior to OMSI telling them it is time to depart (e.g. closing front door while back door(s) still used by disembarking passengers). Due to OMSI limitations this feature may not always work at intended though; a detailed view of the pros and cons is given here.

14 And even for that minority this is a rare sight, due to generally being illegal and considered "a thing of the past".

15 This case is distinct from the case of departing (releasing the stop brake) prior to the front door having (fully) closed.

- There are hardly any passengers on board (static criterion), and in particular none that are either standing or sitting[16] in the first row.

Naturally the front door must also be decoupled[17, 18] from the stop brake for the preference to be applicable. The front door is then left open until a) the vehicle arrives at the next scheduled stop, or b) the environmental constraints no longer hold.

Final notes

- Eager closing due to environmental constraints is static behavior.
- Eager opening due to environmental conditions, as well as eager closing due to lack of passenger flow and timetable constraints, are dynamic.
- Lazy opening and closing are inherently dynamic, since they are mandated by the game.
- Door adjustment delays are as usual dynamic.
- Lazy opening occurs (much) faster than eager opening.
- Closing generally takes longer at prolonged stops. It also takes longer depending on how full the vehicle is (evaluated statically by each vehicle), whether there are standing passengers or passengers seated close to the front door[16, 19], or passengers still walking toward their seat of preference at the original time of departure.
- The decision to leave the front door open at departure is dynamic.

2.4. Stop brake switch

Activation at scheduled stops

Upon arrival at a scheduled stop, vehicles will manually engage the stop brake via the stop brake ("20-h") switch, if either:

- They intend to open their front, yet not their back door(s), and the front door is *not coupled* to the stop brake; or
- The stop is non-prolonged, and/or the engine is, according to OMSI, *not*[20] to be turned off, and the vehicle remains stopped for longer than a few seconds.

Activation at unscheduled stops

A vehicle is said to be at an *unscheduled stop* whenever, for whatever reason, it has to slow down to a standstill without being located within the docking distance of a stop cube on its schedule; e.g. when waiting on a red traffic light. The vehicle may then toggle its stop brake switch if stopped for long enough. How long that might take, and with what frequency the stop brake will be employed, depends

16 Only for *user-focused* AI vehicles; OMSI does not communicate seat occupation status to *unfocused* vehicles, and hence this feature will not work as intended there.

17 That is, opening of the front door will not automatically cause stop brake engagement.

18 See also Appendix B / configuration.

19 And hence greater care is needed by the driver to ensure the front door has fully closed prior to departure.

20 Otherwise the handbrake would too ultimately become engaged, mitigating the need to also engage the stop brake.

on subjective preference and random variation. Regardless the significance of the preference, there will be times when a vehicle does not engage the stop brake, no matter the unscheduled stop's duration.

Switch deactivation

Vehicles will reset the stop brake switch, if previously engaged, just before departing. At unscheduled stops they may also reset it earlier (they may quickly toggle it from "off" to "on" and back to "off").

Closing notes

Stop brake switch activation constitutes static behavior when occurring in the context of front door opening; no vehicle will ever open its front door without engaging its stop brake. All other cases of behavior are dynamic, as are adjustment delays.

2.5. Scheduled stop departure

Stop overlap pseudo-departure

On some maps there are stop cubes that are so close to one another that their configured docking distances overlap, effectively causing the same path segment to technically "belong to" two or more cubes. This manifests more frequently on terminal/initial cube pairs, such as at U-Bhf. Ruhleben on the standard Spandau map. Standard AI vehicles will in such cases "appear to depart"[21], only to instantly "appear to arrive" anew, without actually having covered any distance in between.

Vehicles of this modification will a) detect this case, and b) communicate departure to OMSI, without c) altering their visible state[22].

Parking maneuvers

Sometimes a bus arrives at a (typically prolonged) stop, only to find part of its path blocked (typically by a preceding vehicle), causing it to stop (and "appear to arrive") prior to reaching the center of the cube's docking distance. When the path finally clears, yet departure time has not yet arrived, OMSI instructs that the vehicle depart *without* actually advancing its timetable to the next stop entry, causing the vehicle to merely correct its parking position (hence the naming of such behavior). Modified vehicles will attempt to detect this case and depart much faster under such conditions, to avoid blocking succeeding traffic for too long.

21 By closing doors, turning their engine back on, etc..

22 Put otherwise, whenever the vehicle arrives at stop *n*, and that stop's corresponding cube overlaps with stops *(n+1)*, *(n+2)*, and *(n+3)*, the vehicle will not "appear to depart" until told by OMSI that departure time from stop *(n+3)* has arrived.

Other factors affecting departure delay

Under regular, non parking maneuver departures, a number of factors influence the overall delay:

- The time of departure is significantly delayed at prolonged[23] stops.
- The time of departure is significantly delayed at stops when the engine was turned off[24].
- The longer doors take to close (see above section), the longer departure gets delayed.
- The departure delay is lower when doors were not opened at all at the particular stop, or when all were closed eagerly before the time of departure.

2.6. Electrics, engine, transmission, and handbrake

All three of these functions are handled by the enhanced buses of this modification in more or less the same fashion as standard AI buses. The primary difference is proper sequencing and random delay of adjustment. Note that all of the behaviors described in this section are static.

Electrics

Electrics are generally dependent on whether the engine is running – when it is (ignition key in position 2 or (momentarily) 3), the electrics are fully on, otherwise (ignition key in position 1) they are fully or partially on[25]. When the engine is not running, however, the AI may need to temporarily fully activate the electrics (by rotating the ignition key to position 2, without proceeding to position 3 to ignite the engine), if the driver window or the hatches require adjustment[26] while stopped. After adjustment of the driver window and/or the hatches has completed, electrics are reset to partial / constrained operational state (ignition key rotated back to position 1).

Engine

The engine is turned off only when so instructed by the game[27]; it is then not reignited until actual departure[28]. Furthermore, the engine is not turned off if the weather is perceived by the vehicle either as extremely[29] cold or extremely hot.

---

23 Particularly in the case of initial stops, passengers might not be able to reach the vehicle in time otherwise (due to its destination display failing to update soon enough, and/or the docking distance being too great and passengers walking at snail's pace, and/or OMSI notoriously tending to spawn additional, "late-arriving" passengers, while at the same time instructing the vehicle to depart).

24 To allow non-instant and hence more realistic adjustment of engine, transmission mode, and handbrake.

25 Depending on vehicle configuration, in terms of individual functions' constraints on power availability.

26 The driver window and/or the hatches may require that the ignition key be at position 2 or greater in order to become operable (refer to Appendix B / configuration). Only then will the AI rotate the ignition key prior to using those functions.

27 Generally this is the case when the effective (scheduled time of departure - actual time of arrival) stop duration is greater than 1 minute.

28 Including the stop overlap pseudo-departure case, explained previously.

Transmission

Transmission is set to ‘N’ whenever the game instructs that the engine be turned off, regardless of whether the engine will be turned off in practice; it is then not reset to ‘D’ until actual departure, and not before the engine has been turned back on[30].

Handbrake

The handbrake is trivially engaged whenever the vehicle arrives at a prolonged stop and is told to turn its engine off, and disengaged at departure time.

Adjustment sequences

Typically, upon prolonged stop arrival, vehicles will first a) engage the handbrake, then b) switch transmission mode to ‘N’, and finally c) turn the engine off, if applicable. At departure the typical adjustment order is the inverse, i.e., a) reigniting the engine, then b) switching transmission to ‘D’, and finally c) disengaging the handbrake. Besides the aforementioned constraints governing engine and transmission dependency, however, adjustment delays are orthogonal to one another, and hence it is not uncommon to witness vehicles performing these three actions in any otherwise permissible order.

2.7. Front door wing locking

A fair amount of vehicles that have a wing locking function (switch) installed for the front door, will employ that function when the environmental conditions are perceived as uncomfortably cold (particularly at nighttime, as well as under wet conditions) or uncomfortably warm[31] (particularly at daytime). Furthermore, most of these vehicles will be inclined to lock the front, rather than the rear wing.

Under hot conditions and during prolonged stops when the engine is turned off, the wing lock will be reset, such that both wings of the front door can (eagerly) be opened for the duration of the stop.

Door wing locking preferences are static. Adjustment delays are dynamic, for the most part[32].

---

29 “Extremely” according to the vehicle’s viewpoint, as always; one vehicle might perceive a temperature of 28 degrees as severely hot, while another might perceive the same value as comfortably warm.

30 This is significant when the need for an unanticipated parking maneuver adjustment arises: Even if the vehicle has not had enough time to shut its engine down and switch transmission mode to ‘N’ since its arrival, it will always gracefully ensure the proper, in terms of ordering, adjustment of both functions back to their initial state, that is, a) if the engine was turned off, but transmission is still in ‘D’, switching transmission to ‘N’ first, then b) reigniting the engine, and lastly c) re-switching transmission to ‘D’.

31 Provided the A/C is in use for cooling.

32 Exception: When the front door is to be opened at a time when adjustment of the wing locking switch is pending, the latter is “lazily” rescheduled to occur *exactly* before the former (the driver wants to open the door and just so “remembers” they ought to flip the wing locking switch first).

2.8. Driver window

The degree at which the driver window is opened is dependent on random, per-vehicle preference, for each of the sets of common perception criteria[10] discussed previously for door and door wing locking. Generally the window will be most lowered under a perception matching { “warm”, “day”, and “dry” }. Under wet conditions, the window will usually be closed, as too will be the case under conditions warranting usage of the A/C for cooling.

As with door wing locking, under conditions perceived as uncomfortably warm, at prolonged stops when the engine is to be turned off, the window will be opened significantly and remain open until the time of departure.

The preferred window state for each set of conditions is static. The adjustment delay is dynamic.

2.9. Hatches

Hatch state[33] is driven in exactly the same manner as driver window state. The condition - preference pairs, however, are orthogonal to those governing the driver window. Most vehicles will be more conservative[34] with their use of hatches than their use of the driver window. Additionally, as opposed to the driver window, hatches are not temporarily opened under uncomfortably warm conditions, at prolonged stops when the engine is turned off.

2.10. Passenger windows

Both the simulated AI driver, as well as AI passengers[35] can manipulate[36] the state of passenger windows. The preferred setting per window is determined similarly (albeit once again remaining orthogonal) to the manner in which the preferred state of the driver window and the hatches are. It is important to note that there are two preferences involved – that of the driver and that of the passenger(s), if any, seated close to each window – which need not necessarily align.

---

33 Due to the range of currently supported vehicles being limited to first-generation Citaros, all hatches can at present only be in two states – open (backwardly) or closed. This function may be further refined, should support for different vehicle types be introduced in the future.

34 That is, fewer vehicles will open hatches under sub-optimal conditions, as opposed to the driver window, which may well remain at least somewhat open under even relatively cold or hot conditions.

35 Only in user-focused vehicles, whether AI- or user-controlled, for reasons explained previously.

36 Assuming the vehicle *does have* folding passenger windows that can be opened. See also Appendix B / configuration.

AI driver preferences

AI drivers will typically want one or two windows in the back[37] of the vehicle to be open under neutral-to-cold temperature ranges, gradually increasing their number until the temperature reaches borderline uncomfortable warm levels. Under uncomfortably warm or wet conditions, they will usually want the windows shut. As an additional measure, most drivers will tend to lock the passenger windows under the extreme ends (cold or hot) of the temperature spectrum; some drivers will however always lock them when not in use.

AI driver-conducted window adjustment

AI drivers will only ever alter the state of passenger windows when a) at a prolonged stop, when the engine ought to be turned off according to OMSI, and b) no passengers are on board[38]. This case of vehicle adjustment is special, in the sense that there is no single switch that can just be flipped to instantly carry out the adjustment; instead, it must appear[39] as if the driver were to exit their cubicle, walk to each window, and adjust its state.

AI passenger preferences

Passengers act in groups, each group comprising the occupied seats[40] that are adjacent to each folding window. A group[41] is formed when the first passenger arrives. Subsequently-arriving passengers attach themselves to previously-formed groups. A group dissolves and instantly reassembles[42] when either a) too many newcomers joined in, or b) when too many of the group's founding passengers have disembarked. A group simply dissolves when all its members have left.

In terms of actual preference, passengers desire the same[43] thing as the driver: Open the window when the environment is perceived as "better-outside-than-inside", and close it otherwise. Unlike the driver, passengers have a third option – to "do nothing", implicitly adopting the driver's preference regarding their window as "fair-enough". Also, unlike the driver, passengers will want to open the windows when outside conditions are uncomfortably warm and the engine is not running (and hence nor the A/C). Of course passengers' behavior is also constrained by the driver having potentially previously decided to lock their window.

---

37 Likely due to the temperature in the rear being higher (due to engine conduction); and/or to limit draft between passenger windows and their own window, under lower temperatures.

38 This ensures there is no "war" between conflicting driver - passenger viewpoints resulting in endless "cycling" – one party opening the window and the other closing it shortly thereafter.

39 Temporally-wise at least, since we can't animate the actual driver figure.

40 Sitting or standing places declared at the `passengercabin.cfg` level, that is.

41 Refer to commentary in `ai_pre_cockpit.osc:ai_pre_cockpit_passengers` for a more detailed discussion on group dynamics.

42 With potentially different opinions, i.e., preferred window state.

43 That is to say, the logic is the same, while the concrete perception may differ.

Closing notes

The driver's preferences are static. Group preferences are static throughout each group's lifetime. Adjustment delays are dynamic.

2.11. Heating and cooling

For heating purposes under a somewhat cold environment, vehicles will rely on either the A/C or the cabin heaters. For heating purposes under a significantly cold environment, vehicles will always use their more effective cabin heaters. Regardless this distinction, vehicles will also increase the driver-specific output temperature and fan speed, while leaving the driver A/C function in passive, engine-assisted mode, while in a heating context.

For cooling purposes under hot, as well as environmental conditions that are both warm and wet, vehicles will rely on the A/C (both passenger and driver functions), adjusting the driver-specific output temperature to a medium-to-cool setting, as well as increasing the driver-specific fan speed.

Lastly, under mostly-comfortable conditions, some vehicles may choose to use the driver A/C in "plain fan" mode, increasing the driver-specific fan speed, while retaining the lowest permissible output temperature, and without otherwise activating either of the two A/C functions.

Heating and cooling preferences[44] are static. Adjustment delays are dynamic.

2.12. Sun blinds

Sun blind state depends on the perceived solar elevation angle[45], as well as the common temperature ranges, the common dryness/wetness dichotomy, and the primitive level of environmental brightness[46]. The probability that a vehicle drag their sun blind(s) down is greatest around midday, particularly when the environment is additionally perceived as warm, dry, and bright. Those vehicles that choose to employ their sun blind(s) in the morning or evening hours, however, will tend to also drag it down further during those periods than at midday. Furthermore, the likelihood of a vehicle using its main (windshield) sun blind is greater than the likelihood of it using its side (driver window) sun blind (where installed).

---

44 Statically-variable preferences: A/C vs. cabin heaters for heating purposes; driver A/C temperature and fan speed combinations for heating and cooling; usage of driver A/C as "plain fan" (and, if so, the fan speed) under generally favorable conditions. The remainder of the aspects discussed are not really preferences, in the sense that they do not vary between vehicles.

45 Here the angle is subjectively classified as either *night*, *morning/evening*, or *midday*.

46 Here brightness is reduced down to a simple dichotomy of *(not) bright*.

Dragging either sun blind down is only permissible[47] when the vehicle is halted at either a scheduled or unscheduled stop, and no other adjustments are scheduled to concurrently occur. Sun blind retraction is allowed as long as no other adjustments are pending.

The preferred position of each sun blind under each set of conditions is static. Adjustment delays are dynamic.

2.13. Wipers

Wiper state depends on the precipitation rate[48] (obviously). Furthermore, it depends on the precipitation type, with rain generally being perceived as "more intense" / "more severe" than snowfall. Lastly, it depends on the common day/night dichotomy, along with primitive environmental brightness, with *brighter* conditions generally translating to precipitation being perceived as "*more* intense"[49].

Only some vehicles will ever employ the "fast" wiper operational mode; most will stop at the "continuously on" speed, regardless the precipitation intensity.

When at prolonged stops, when told by the game to turn their engine off, most vehicles will deactivate their wipers until departure. Some will instead leave them running[50], possibly at a lower mode of operation.

The preferred wiper mode under each set of conditions is static. Adjustment delays are dynamic, and shorter when wipers are to be enabled or transitioned to a greater speed, than when they are to be transitioned to a lower speed or disabled.

2.14. Indicators

The main contexts of indicator adjustment[51] currently implemented are: a) Adjustment when arriving at a stop; b) adjustment when departing from a stop; and c) adjustment en route. There are also the

47 Because the driver would generally need both their hands free to drag down a blind, as well as be able to momentarily leave their seat (mandating that at least the stop brake be active). Electric sun blinds are currently unsupported.

48 For wiper-centric perception, precipitation specifically is classified as *very light* (wipers only manually toggled on/off when windshield gets wet enough), *light* (wipers set to "interval"), *moderate* (wipers set to "continuously on"), and *high* (wipers set to "continuously on" or "fast", depending on per-vehicle preference).

49 Because particularly under rainfall, a wet windshield is more "annoying" / vision-impairing when reflecting ambient light, which is of course of greater intensity during the day.

50 Whether the wipers actually stay on then depends on whether a) the engine happens to remain on, or otherwise b) whether the wipers remain operable under reduced electrics state (ignition key at position 1). Currently no special accommodations are made for (b), i.e., electrics will not be turned fully on (ignition key at position 2) solely to enable wiper operation in AI vehicles desiring it.

51 Indicator adjustment here refers to the action of a) enabling (off → left/right), b) disabling (left/right → off), or c) inverting (left ↔ right) the indicator mode.

secondary cases of d) deactivating indicators at prolonged stops, when the game requests that the engine be shut down, and e) passively having the indicators deactivated due to having exceeded the "critical" steering angle towards the corresponding direction.

In all cases it is possible that the simulated AI driver "forget" / "neglect" to adjust their indicator mode. It is also possible that the adjustment is carried out after a shorter or longer delay.

On a least- to most-likely-to-forget scale, the various cases can be ordered as follows:

- (Least likely) (a) At scheduled stop arrival.
- (c) En route, with activation or adjustment being less likely to be neglected than (timely) deactivation.
- (b) At scheduled stop departure[52], most vehicles will most of the time opt to *invert* their indicators[53]; some may lazily just turn them off instead.
- (e) In the event of automatic indicator deactivation due to steering, vehicles will be somewhat prone to lazily leaving them off, even if not yet having completed that turn, or having exited that junction.
- (Most likely) (a) Deactivation[54] at prolonged / engine-off stops.

When indicator adjustment is forgotten, the vehicle will most likely remember[55] to correct them the next time a change of indicator setting gets communicated to it by the game.

Indicator "forgetfulness" and adjustment delays are dynamic. While certain vehicles will on average be more prone to forgetting to (timely) adjust their indicators, none are guaranteed to *never* or *always* act in the same way.

2.15. Dirtiness

Conventional AI vehicles usually either do not ever get dirty, or only begin accumulating dirt when within the user's focus for a while. For added realism, this modification allows its vehicles, while in AI mode, to both spawn in an already dirty state, as well as get dirtier over time, even when not explicitly focused by the user – in fact, even when outside the user's loaded tiles altogether. This adds to the

52 Note that OMSI's (the path segment's, to be more precise) opinion on the matter is effectively ignored in this case; as long as the path segment *before* the stop mandated the proper behavior, the behavior at departure will be automatically and correctly deduced. Exception: When the path segment *just after* a stop mandates anything other than indicator deactivation, its instruction will still be obeyed by vehicles (not taking forgetfulness into account).

53 In other words, if they indicated right at arrival, they will indicate left at departure.

54 Naturally, in vehicles where indicators are only functional when electrics are fully activated (ignition key at position 2 or greater), forgetful behavior will not manifest, unless the engine is left running at that stop.

55 Exception: When there are several adjacent path segments that all mandate the same indicator setting (e.g., two subsequent right turns), and the vehicle forgets to adjust its indicators the first time around, it will also keep forgetting their adjustment every subsequent time, until finally told to deactivate them. This is fine as long as all path segments represent *the same* junction / curve / turn – when this is not the case, i.e., when such segments stand for *consecutive, yet independent* junctions / curves / turns, repetitive neglectful behavior exhibited by vehicles will appear highly unrealistic. Unfortunately this is a known limitation of the current algorithm.

illusion that there is a *past* – that the world didn't just instantaneously come into existence when the OMSI situation was loaded.

As with user-controlled vehicles, AI vehicles will generally be or become dirtier in proportion to the weather conditions, particularly precipitation; ultimately, the wetter the streets, the faster dirt will accumulate, and the worse a shape will AI vehicle's exteriors end up in. Unlike user mode, however, there is also an additional, random per-vehicle factor[56] contributing to the dirt accumulation rate and maximum dirtiness in AI mode, allowing – most notably under favorable / dry weather conditions – some vehicles to get dirtier than others.

2.16. Non-functional requirements

Resilience to use interference

With the exception of a) opening the folding passenger windows, if present and unlocked, and b) pushing the stop request buttons, no other[57] cockpit triggers / mouse events are made available to the observing user when the vehicle is AI-controlled. *Accidentally* interfering with the AI work-flow and causing the vehicle to remain stuck indefinitely should therefore be impossible[58].

Resilience to OMSI interference

OMSI is obnoxious at de-/re-spawning/teleporting AI buses at will, without ever explicitly notifying them about the change in context. Significant measures have been taken to ensure that vehicles can detect such anomalous transitions and continue operation as gracefully[59, 60] as possible.

---

56 Seeking to approximate all those external influencing factors that vary per vehicle and cannot be directly simulated; e.g. the streets of a certain district served by one bus being in a worse condition than those of another, served by a different bus; or one driver having a more aggressive driving style than another.

57 With the exception of still being able to affect the steering wheel, as well as the throttle and brake pedals via mouse or gaming controller, which is unintentional – OMSI appears to use its own internal "triggers" for such primitive input.

58 Certain exceptional edge cases currently uncovered: a) When crashing a user- into an AI-controlled bus; b) when allowing the AI to take over a severely damaged vehicle. Furthermore, when the user *purposely* wants to cause the AI a headache, they will always find a way, including the indirect one of modifying the scripts.

59 For example, suppose an AI vehicle was last active at a prolonged stop, with a shut-down engine and open doors. Normally that vehicle would require a significant amount of time – let's say half a minute on average – to depart in a "human-like" fashion, as explained in previous sections. If suddenly teleported to the middle of a busy junction though, the vehicle would have to resume operation *instantly*, such that the user can't notice the discrepancy – not even implicitly due to other AI traffic being blocked.

60 As a second example, suppose an AI vehicle was last active at a point in time where the weather changed from "rainy" to "perfectly sunny". As explained in previous sections, after a bit of time the observer would expect that this very vehicle would have undergone certain changes in state, e.g. having switched its lights off or having opened some windows. Sadly, the notion of time isn't continuous for AI vehicles – they only "exist" for as long as they remain on the same loaded tiles as their observing user. Therefore, adjustments of minutes would likely take hours to occur, or fail to ever manifest at all. To compensate, AI vehicles have been made aware of such "temporal anomalies", so as to instantly adjust their state if need be, and such that their observer can't tell the difference, i.e., that the particular AI vehicle was only loaded for a total of a minute and a half over the span of the 3 hours the user spent driving their own bus across the map.

Deactivation of unsupported functions

When the AI takes over a previously user-controlled vehicle, it will deactivate or otherwise invalidate any functions it does not "natively understand"[61], thereby effectively preventing non-aesthetically-pleasing or erroneous side effects.

61 For example the AI is capable of deactivating the retarder switch, detect and cancel-out out a transmission in 'R' (by switching back to 'N'), drag the driver cubicle door / ramp / doors opened via emergency valves back closed, etc..

3. Other significant changes

3.1. Lighting

The lighting subsystem has undergone significant refinement as part of this work.

Almost all[62] light sources, both on the exterior and interior, up to and including marker lights and the lights of the rear license plate, have become independently operable in one way or another. This enables individual light sources to have and/or produce more realistic illumination effects, to individually affect circuit voltage, and to individually become defunct under different conditions.

Light sources can already be defunct when a vehicle first gets spawned, including vehicles initially spawned in AI mode. Light sources can also become defunct subsequently, either due to naturally reaching their end-of-life, or due to receiving collision damage. The overall probability and frequency of either event occurring depends on a) the type[63] of source, b) its significance[64] and where[65] exactly it is installed on the vehicle's body, c) the vehicle's age[66], d) the user's maintenance setting[67], and of course e) if applicable, collision severity. It should also be noted that even when a collision does not produce visibly damaged lights, lights in the proximity of the collision's coordinates will still be *strained* as an implicit result, negatively affecting their lifetime expectancy.

Last but not least, the circuit-level constraints governing activation of light source (groupings) have been revisited as well. Particularly when the engine is not running and the battery is weak, certain non-essential lights will be *throttled* before the battery depletes, as a precautionary measure to conserve energy.

Some further changes are enumerated in the changelog section.

---

62 With the exception of matrix-integrated light sources, as well as dashboard lighting and function indicators, which have not yet been integrated with the revamped lighting subsystem.

63 For example, the lifetime of a LED source is generally greater than that of a fluorescent, in turn being generally greater than that of a halogen one.

64 For example, during maintenance, a headlight is more likely to be replaced on time (prior to burning out) than a marker light.

65 Both in terms of significance and collision susceptibility. For example, an orange marker light, *A*, located towards the middle of the vehicle, is statistically less prone to being damaged as a result of "micro-collisions" (pavement "scratches") than another orange marker, *B*, located closer to the vehicle's head or tail. Furthermore, *A* might be considered of greater contributing value to safety, and maintenance will be more likely to identify and replace it on time. For both reasons, the user will generally be more likely to witness light *B* in a defunct state, than light *A*.

66 Older vehicles will have batteries and other components of their electrical circuitry in a sub-optimal state compared to younger vehicles, in turn (slightly) negatively impacting performance (lifetime) of connected light sources.

67 Only for user-spawned vehicles, and only while user-controlled; for AI-spawned/-controlled vehicles a random maintenance factor – typically "very good" – is instead chosen, due to OMSI then disregarding / not communicating to such vehicles the user's setting (but instead always assigning a factor of "infinite").

3.2. Heating and cooling

The implementation of the heating and cooling subsystem has been replaced with a different one, yielding a number of improvements. This section enumerates some key differences between the current and original implementation.

All heating and cooling functions now undergo non-instant adjustment, i.e., feature “inertia”. The length of an adjustment period depends on the kind of function, the difference between the current and target temperature and humidity, the target state (function activation vs. deactivation), and random variation. Simply put, the user should generally anticipate that flipping a switch will *not* produce a visible change in sound, temperature and humidity until several seconds or even minutes have elapsed.

A new passive function, coined “greenhouse-like effect”, has been introduced. Depending on insolation and particularly during the summer, the inside temperature can rise significantly above the ambient (outside) temperature. This effect can to some extent be mitigated by keeping windows, hatches, and doors open as much as possible, or using the A/C functions.

The passenger A/C function has become more effective in cooling[68] mode, while its effectiveness has been somewhat decreased in heating mode. For heating purposes, the cabin heaters remain a much more effective alternative. As usual, the passenger A/C’s output and allowed (minimum, maximum) target temperatures cannot be altered by the user[69]. Its fan speed will automatically adjust depending on the difference between the current and target cabin temperature.

The driver A/C function[70] has been greatly reduced in effectiveness – after all it is supposed to heat or cool the driver cubicle and not the entirety of the vehicle. Thus, even at maximum temperature and fan settings, the driver A/C will not appear to significantly influence the overall[71] vehicle temperature. The driver A/C is totally ineffective[72] for heating purposes – leaving it in “passive”, engine-assisted mode is a much better idea when intending to heat the driver’s place.

Both A/C functions are implicitly coupled to a humidity management ((de-)humidification) function. That function will automatically activate whenever a) any A/C function is active, and b) the cabin air has gotten extremely dry or moist[73]. The user can also explicitly activate it via the “heat-or-frost”

68 The A/C functions will by default perform heating when the temperature inside is below 19 degrees, and cooling when the temperature inside exceeds 22 degrees.

69 They can however be adjusted via the many constants in `uchill_consts.txt`.

70 Including the sub-function where it operates in engine-assisted, as opposed to A/C-based mode.

71 In a future version we might further emphasize the fact by simulating different “thermometers” for a) the driver cubicle and b) the reset of the cabin. This is a non-trivial exercise, however, and OMSI is anything but helpful in that regard.

72 Exception: For humidity management purposes (humidifying or dehumidifying the vehicle), leaving the driver A/C active actually *does* make sense.

73 In terms of relative humidity this translates to, approximately, below 15% or above 85%.

button[74] for optimal results. When humidity management is active, the “air origin” (fresh vs. recycled) mode of both A/C functions will transition to “recycled”.

Both A/C functions also feature an alternative, implicit, “passive” / “eco” mode, which is automatically chosen when the temperature on the outside is already (much) more comfortable than the temperature on the inside. The A/C functions will then effectively operate as plain fans, just allowing air from the outside to flow inside, until equilibrium is achieved.

Whenever a function is explicitly activated, yet constraints[75] for its activation do not hold, or has been implicitly / automatically enabled by the system, the dashboard indicator of the corresponding button or switch, where available, will blink.

For further details on heating and cooling functions, the user is encouraged to refer to the external documentation of the UCHill project.

74 Usually to be found on the classic 4-button side panel.

75 For example, A/C active but engine off; or humidity management active but humidity already within comfortable interval.

A. Triggers

This section captures triggers (mouse- and keyboard-events) that have been newly-introduced, renamed, and/or semantically altered as part of this work.

Name	Description
`automatic_[1|2|3]`	These triggeres respectively select the constrained *1* ("up-to-first-speed"), *2* ("up-to-second-speed"), and *3* ("up-to-third-speed") transmission panel modes, when the effective value of the `cp_cfg_xxx_has_additional_transmission_mode_selectors` group of settings is 1. For any of the three buttons to be push-able, the currently selected transmission mode must not be *R* (reverse). See also `antrieb_cfg_xxx_standstill_non_neutral_transmission_mode_update_requires_braking` setting.
`cp_fog_light_mode_increase_or_disable`	Increases or disables the vehicle's fog lights' mode, transitioning as follows: *off [→ head* fog lights] → head* + tail fog lights → off* * If available, depending on the effective value of the `lights_cfg_xxx_has_head_fog_lights` group of settings.
`cp_fog_light_mode_decrease`	Decreases the vehicle's fog lights' mode until completely disabled, transitioning as follows: *head* + tail fog lights [→ head* fog lights] → off* * If available, depending on the effective value of the `lights_cfg_xxx_has_head_fog_lights` group of settings.
`cp_passenger_window_[1|2]_[l|r]_lock`	In vehicles having passenger windows, this trigger locks or unlocks the corresponding window (labeled beginning from the vehicle's front), *iff* a) the effective value of the `cp_cfg_xxx_has_openable_passenger_windows` group of settings is 1; and b) the window is closed. Note: Provided `cp_cfg_xxx_has_openable_passenger_windows` is 1, when a *user vehicle* is initially spawned, its passenger windows are all always initially *unlocked* (due to there not currently being a mouse-event that one could conveniently employ to unlock them otherwise).
`cp_passenger_window_[1|2]_[l|r]_open`	In vehicles having passenger windows, this trigger opens or closes the corresponding window (labeled beginning from the vehicle's front), *iff* a) `cp_cfg_xxx_has_openable_passenger_windows` is 1; and b) the window is *unlocked*. Note: *Unless locked*, AI passengers and/or, if applicable, the AI driver, may still adjust (open, close, or, additionally, in the case of the driver, lock) the window, regardless of your opinion.
`cp_not_aus_sw_toggle`	Toggles on the dashboard's red "NOT-AUS" switch, releasing the stop brake, bypassing all constraints that would normally prevent release. These constraints include any open doors (front door depending on `door_cfg_xxx_front_door_decoupled_from_stop_brake`), whether the rear door

	wheelchair ramp is currently unfolded, whether the vehicle is currently knelt or in the process of kneeling down, and whether ECAS leveling is underway. For stop brake override to succeed, *while the switch is still flipped / triggered*, the throttle pedal must additionally be pushed. If at the same time the manual stop brake engagement (“20-h”) switch is also active, then it will still be honored, i.e., the stop brake override request will be ignored. If override succeeds, and the vehicle was previously knelt down, it will automatically rise back up, independent of door state. Furthermore, if override succeeds, the VDV will display a warning for as long as any violation of a constraint normally precluding stop brake release remains in effect, i.e., until all stop-brake-coupled doors have closed, the wheelchair ramp has been folded, the vehicle has knelt back up, and ECAS leveling has concluded.
`bus_doorfront0_ext`	Toggles the external opening button for the front door, acting in the exact same way as the “standard” `bus_doorfront0` trigger, except that electrics need not be switched on for door adjustment to succeed.
`lights_ai_test_toggle`	Initiates the diagnostic mode of the AI lighting script, unless already running. Refer to Appendix C for details.
`kw_wipermode_[up|down]`	These triggers have been modified to behave equivalently to `cp_fog_light_mode_increase_or_disable` and `cp_fog_light_mode_decrease`; i.e., the former progressively increasing the wipers’ mode until “maxed”, then disabling them, and starting over; while the second progressively decreasing the wipers’ mode until completely disabled.

B. Configuration

Introduction

The vehicles of this modification employ an extended configuration scheme. This mechanism seeks to:

- Promote, within the confines of what is reasonable and practical, the reuse of the same script- and const-files across different vehicles, variants, and models, thereby reducing the burden, imposed on both ourselves and users, of unnecessary maintenance of almost identical copies of such files for each vehicle.
- Provide sensible configuration defaults, that are expected to apply most of the times, such that these settings need not be specified anew for each vehicle, unless actually differing from the norm.
- Allow for flexibility in overriding the defaults as needed, both statically (const- and .cti-files), and dynamically (in-game).

Structure

Configuration is organized in a hierarchy, with lower layers being more general and expected to apply more often, and higher layers being more specific, representing slight differences one tends to witness between individual vehicles, even when of the exact same model and variant. Generally speaking, higher-level configuration, when specified, "overrides", i.e., *takes precedence over*, lower-level configuration.

Layer 0: "Default-defaults"

At the bottom-most configuration layer, *L0*, scripts rely on hard-coded defaults / "fallback" settings, always assumed valid. These settings remain in effect until valid overriding configuration at any higher layer gets specified.

Layer 1: Per-model-/-variant-/-component-defaults

At the next layer, *L1*, configuration is given by const-file entries. Such entries, if found valid, override the L0-defaults. L1-configuration is the proper place for specifying configuration that applies to a wide array of vehicles, all sharing a particular component / subsystem, by virtue of having a shared const-file; for example all vehicles equipped with the same engine or gearbox, or the same door scheme. Lastly, L1-configuration also takes over the role of the next higher layer, *L2*, for vehicles using the default, unnamed repaint (to which normally no .cti-file applies).

There are two (plus one) kinds of constants integrated with L1-configuration:

`<subsystem>_cfg_default_<setting>`
These are actual settings. The acceptable set or range of values of each depends on the nature of the task fulfilled by the setting, as documented in this section.

`<subsystem>_cfg_no_cti_override_<setting>`
These are complementary "override switches", one per setting. Each can either be set to 0 ("disabled") or 1 ("enabled"). When a switch is enabled, it prevents the corresponding L2-setting, if specified, from

overriding the one at L1. This is intended for cases where one needs a quick provisional way of trying out a particular option, without paying the price of first having to edit all apposite L2 configuration (.cti-files, potentially dozens).

`global_cfg_no_cti_overrides`
This is a "global" alternative to the individual `<subsystem>_cfg_no_cti_override_<setting>` switches discussed above, to be found in the `global_consts_xxx.txt` const-file referenced by each .bus-file. When enabled, the resulting behavior is equivalent to every individual override switch having been enabled.

Layer 2: Repaint-specific overrides

At this layer, *L2*, .cti-file `[setvar]`s are processed. The configuration variables on this layer follow the naming scheme **`<subsystem>_cfg_cti_<setting>`**, with the <*subsystem*> and <*setting*> components being identical to their L1 counterparts. When the assigned value differs from the effective default (L1 or L0), it takes precedence – unless the corresponding L1 "override switch" is enabled.

Layer 3: In-game overrides

At the last layer, *L3*, there are variables following the naming scheme **`<subsystem>_cfg_on_demand_<setting>`**, with, as one would expect, the <*subsystem*> and <*setting*> components remaining, once again, identical to their counterparts at the preceding two layers. These variables are intended to allow the user to change the configuration "on-demand", after already having spawned their vehicle. When the assigned value differs from the currently effective setting (L2/L1/L0, or previous L3-override), it takes precedence.

At present, while functional in terms of implementation, these variables cannot be directly exploited by users, given that no convenient in-game interface for their reassignment, such as the *Citybus O405* add-on's "magic handbook", or the *MAN Stadtbusfamilie* add-on's configuration display, has thus far been made available. This may change in a future version. Until then, users familiar with the scripting language may author custom triggers for applying L3 configuration changes as desired.

Validation

Values assigned to constants and variables participating in the extended configuration scheme discussed herein, are subject to austere validation. Whenever an assigned value is deemed invalid, i.e., not lying within the set or range of values explicitly declared as acceptable for the corresponding setting, it is disregarded, in favor of the value specified at the next lowest layer. This "fail-fast" behavior is normally a good thing, as it facilitates predictable script behavior, allowing bugs to reach our attention sooner. Nonetheless, it may not always align with the conventional way, familiar to users, in which variable assignments are processed by scripts, where, e.g., the value of zero implies "off", while *any other* (non-zero) value implies "on". To minimize potential confusion, solely values documented as valid in this section should be assigned.

Limitations

In an ideal world, we would want every single constant to become integrated with the new, extended configuration scheme. This is however practically impossible, both because of the sheer number of constants there is, many of which are relied upon by preexisting code, which would drive the

complexity and maintainability cost through the roof, as well as the fact that many of these constants are so fine-grained that few, if any, users would care about having the ability of customizing such aspects per-repaint or in-game. Thus a line had to be drawn somewhere.

As things currently stand, we are only interested in integrating cases of configuration that fulfill the following criteria:

- Variation must occur within the bounds of the CSB vehicle fleet. This means that there is no desire to implement support for e.g. exotic door schemes or passenger information systems that no CSB vehicles use.
- Variation must have limited functional, and ideally solely aesthetic, impact. While we *could* in principle support replacement of entire complex systems, the cost, including the implicit one of declining performance (as the result of then also having to support multiple alternative texture- and/or sound-sets), would fail to justify the benefit. This means that alternative destination displays, engines, or gearboxes, are out of scope.
- Variation must be perceivable by, and potentially of value to, users. For example, people generally care about being able to configure whether there are fog lights mounted on the head bumper; but they usually do not care about being able to specify, say, the Watts the respective bulbs consume, or how many hours they are expected to last. Hence, the former case is a suitable candidate for inclusion, as opposed to the latter ones, which are fine as-is, i.e., as plain constants or hard-coded values.

Users in need of flexibility beyond what the configuration options are capable of offering, are obviously welcome to experiment with altering the values of the plethora of other regular, non-integrated constants, as well as hacking on the scripts directly. As both a disclaimer and a general rule, however, it should be noted that we will be reluctant to provide support to such “power users”, given that they tend to be in the minority, and due to their use cases likely being untested, requiring, on average, more of our limited time and manpower to serve, compared to the majority, adhering to the typical / as-intended / “plain vanilla” use case.

B.1. Cockpit / dashboard

Subsystem name prefix	`cp`		
L1 const-file	`cockpit_constfile.txt`		
Setting name suffix	**Description**	**Constraints**	**L0-default**
`has_additional_transmission_mode_selectors`	Specifies whether the transmission mode selection panel includes the additional constrained modes (‘1’/’2’/’3’) or not. This is a replacement for the `cockpit_gear_D_only` constant. When disabled, triggers `automatic_[1|2|3]` will not be available.	Boolean[76]	0
`has_engaging_neutral_transm`	Specifies whether the *N* transmission mode selector resets to	Boolean	1

76 So-called *boolean* settings, also known as “switches” or “flags”, only accept a 0 or a 1, with 0 implying “false” / “not applicable” / “disabled”, and 1, implying “true” / “applicable” / “enabled”, depending on the setting’s nature, as per its name and description.

`ission_mode_selector`	its original position once no longer triggered, or remains pushed-down. This is a partial replacement for the `cockpit_gangwahltaster_alt` constant. 0 / “false” tends to be the case for *Voith*, while 1 / “true” for *ZF* transmission mode selector panels.		
`transmission_mode_selector_blinking_mode_malfunction`	Specifies whether and which transmission mode selectors shall blink in the event of transmission malfunction. • 0: No blinking; the indicator corresponding to the transmission mode currently effective[77] is to remain constantly on, as is normally the case. • 1: The indicator of the currently effective mode shall blink. • 2: All indicators shall blink synchronously. • 3: All indicators shall blink sequentially (‘1’, then ‘2’, …, then ‘R’). The blinking interval is given by the `transmission_mode_selector_blinking_interval_malfunction` setting.	{ 0, 1, 2, 3 }	2
`transmission_mode_selector_blinking_mode_desync`	Specifies whether and which transmission mode selectors shall blink when the transmission mode currently in effect does not (and cannot) align with the one requested (via the dashboard). • 0: No blinking; the indicator corresponding to the transmission mode currently *effective* is to remain constantly on, as would normally be the case. • 1: The indicator of the currently *selected* mode shall blink. • 2: (1) and (2) combined; *effective* mode constantly illuminated; *selected* mode blinking. Note that this setting is disregarded in the event of transmission malfunction – even if the `transmission_mode_selector_blinking_mode_malfunction` setting has been disabled. The blinking interval is given by the `transmission_mode_selector_blinking_interval_desync` setting. Depending on gearbox-specific characteristics, blinking may occur in the following cases: • When the engine is not running and any mode	{ 0, 1, 2 }	2

77 a) *Selected* vs b) *effective* vs c) *effective-overriding* transmission mode: (a) is always equal to the mode represented by one of the dashboard selectors, when pushed. (b) stands for the last such requested mode that was successfully acknowledged by the gearbox, i.e., when all constraints (e.g., having to brake at standstill) governing mode update were satisfied. Thus, (b) either equals (a), or a past value of (a). Lastly, (c) is either equal to the effective mode, or – usually under abnormal operational conditions – a different one deemed more appropriate by the gearbox; for example, when ‘R’ is effective, yet the vehicle actually happens to be moving forward, the gearbox may override that mode, internally transitioning to the equivalent of ‘N’. Unlike (a) and (b), (c) is independent of both of the former modes, including past values of theirs. Gear engagements and disengagements ultimately performed by the gearbox are a function of (b) and (c). Transmission mode selector blinking, in contrast, is driven by all three modes, with (b) and (c) being used to determine the constantly illuminated selector, and (a) the blinking one, depending on the configured settings.

	besides ‘N’ is selected. • When at standstill, an update from ‘N’ to any other mode is attempted, while not braking (depending on the value assigned to the `antrieb_cfg_xxx_standstill_non_neutral_transmission_mode_update_requires_braking` set of settings). • When, while moving forwards and faster than a threshold, an update from ‘N’ to ‘R’ is requested. Likewise, when ‘R’ is *effective*, yet the vehicle actually moves forward (normally only possible when facing downhill and neither throttling nor braking) and exceeds that same threshold. • When an update from ‘N’ to a constrained mode (‘1’/’2’/’3’) is requested, while moving at a velocity greater than the maximum velocity permissible in the highest speed attainable under the requested mode. • When, while in a constrained mode (‘1’/’2’/’3’), the gearbox enforces an upshift above the normally maximum attainable speed of the mode in effect, to prevent damage (normally only possible when purposely accelerating downhill).		
`transmission_mode_selector_blinking_interval_malfunction`	Specifies the transmission mode selector blinking interval in the malfunction case. Disregarded when `transmission_mode_selector_blinking_mode_malfunction` is zero.	$0 < x \leq 5$ (sec)	0.25
`transmission_mode_selector_blinking_interval_desync`	Specifies the transmission mode selector blinking interval in the case of the effective transmission mode not matching the requested one. Disregarded when `transmission_mode_selector_blinking_mode_desync` is zero.	$0 < x \leq 5$ (sec)	0.5
`has_door_wing_lock_switch`	Specifies whether the dashboard offers a front door wing lock switch.	Boolean	1
`has_hatch_switch`	Specifies whether the dashboard has a switch or set of switches installed for roof hatch control. Note that, depending on the vehicle, this setting will not as of yet magically hide the actual hatches as well, if present. It simply is intended for use with those CSB vehicles whose real-world counterparts either do not have any (electrically-operated) hatches, or have had, for whatever reason, the switch or switches for controlling them removed.	Boolean	1
`has_openable_passenger_windows`	Specifies whether the vehicle has openable passenger windows. When disabled, the `cp_passenger_window_xxx_[lock|open]` triggers will not be available. Neither the user nor the AI will be able to interact with the windows. Note that, depending on the vehicle, this setting will not as of yet magically hide the actual passenger windows / frames / handles as well, if present. It simply is intended for use with those CSB vehicles whose real-world counterparts either do not have any such windows, or have them permanently locked or sealed off.	Boolean	1

`has_blinking_hazard_indicator_switch`	Specifies whether the hazard direction indicator mode switch blinks when active.	Boolean	0
`hatches_require_full_elec`	Specifies to what degree roof hatch control depends on the state of electrics. 0 / “false” means that hatch adjustment requires that the ignition key be at position 1 (“electrics on” / “ACC”) or greater. 1 / “true” means that hatch adjustment requires that the ignition key be at position 2 (“pre-ignition” / “ON”) or greater.	Boolean	1
`driver_window_requires_full_elec`	Specifies to what degree driver window control depends on the state of electrics. 0 / “false” means that driver window adjustment requires that the ignition key be at position 1 (“electrics on” / “ACC”) or greater. 1 / “true” means that driver window adjustment requires that the ignition key be at position 2 (“pre-ignition” / “ON”) or greater.	Boolean	1

B.2. Transmission

Subsystem name prefix	`antrieb`		
L1 const-file	`antrieb_constfile_xxx.txt`		
Setting name suffix	**Description**	**Constraints**	**L0-default**
`standstill_non_neutral_transmission_mode_update_requires_braking`	Specifies whether, while at standstill, braking is required in order for the gearbox to acknowledge a transmission mode update request from ‘N’ to any other mode. Note: This restriction only affects transmission mode update when the vehicle is stationary; it is disregarded when moving. Specifically, when moving forward, the selection of any forward-driving, or of the neutral mode, will always be acknowledged, whether braking or not. Likewise, when reversing, the selection of either ‘N’ or ‘R’ will be acknowledged, irrespective of the brake pedal’s angle, too.	Boolean	1

B.3. Doors

Subsystem name prefix	`door`		
L1 const-file	`door_constfile_xxx.txt`		
Setting name suffix	**Description**	**Constraints**	**L0-default**
`front_door_decoupled_from_stop_brake`	Specifies whether the stop brake is automatically engaged upon opening the front door via its dashboard button. This is a replacement for the traditional `Door_HST_Bremse_Aktiv` constant. One should however pay attention to the inverted semantics of its values. Enabling this setting may (rarely) allow the AI to drive with open front doors, as explained in 2.3.	Boolean	0

B.4. Lighting

Subsystem name prefix	`lights`		
L1 const-file	`lights_constfile.txt`		
Setting name suffix	**Description**	**Constraints**	**L0-default**
`has_head_fog_lights`	Specifies whether there are head fog lights installed.	Boolean	0
`has_drl`	Specifies whether there are DRL installed. Note that DRL support is currently only implemented at the script level, for future use. To get an actual, “DRL-capable” vehicle at present, one must additionally, at the `model.cfg` level: • Introduce the actual (2x) DRL light unit `[mesh]`es. • Define accompanying visual effects via any of the `[matl_nightmap]`, `[spotlight]`, and `[light_enh[_2]]` directives, given the state of the `lights_drl_[l|r]_unit` variables.	Boolean	0
`has_second_reverse_and_tail_fog_light`	Specifies whether there is an additional pair of one reverse and one tail fog light installed at the rear. Note that this setting is currently only implemented at the script level, for future use. To integrate the additional light sources, one must additionally, at the `model.cfg` level: • Introduce the relevant light unit `[mesh]`es. • Define accompanying visual effects via any of the `[matl_nightmap]`, `[spotlight]`, and `[light_enh[_2]]` directives, given the state of the variables: ○ `lights_rueckfahr_l_bulb` and/or	Boolean	0

	lights_rueckfahr_l_matl_mode, for extra reverse light. ○ lights_nebelschluss_r_bulb and/or lights_nebelschluss_r_matl_mode, for extra tail fog light.		
indicating_left_or_right_requires_full_elec	Specifies to what degree the activation of direction indicators in simple “left-or-right” mode depends on the state of electrics. 0 / “false” means that activation requires that the ignition key be at position 1 (“electrics on” / “ACC”) or greater. 1 / “true” means that activation requires that the ignition key be at position 2 (“pre-ignition” / “ON”) or greater.	Boolean	1
int_lights_level_2_requires_running_engine	Specifies to what degree the activation of the “full” mode of the main interior passenger light sources depends on the state of the engine. 0 / “false” means that activation requires that either the engine be running, or the ignition key be at position 2 (“pre-ignition” / “ON”) or greater. 1 / “true” means that activation requires that the engine be running.	Boolean	1

B.5. Wipers

Subsystem name prefix	wiper		
L1 const-file	wiper_constfile.txt		
Setting name suffix	**Description**	**Constraints**	**L0-default**
wipers_require_full_elec	Specifies to what degree wiper usage depends on the state of electrics. 0 / “false” means that wiper usage requires that the ignition key be at position 1 (“electrics on” / “ACC”) or greater. 1 / “true” means that wiper usage requires that the ignition key be at position 2 (“pre-ignition” / “ON”) or greater.	Boolean	1

B.6. Heating and cooling

Subsystem name prefix	`uchill`		
L1 const-file	`uchill_consts.txt`		
Setting name suffix	**Description**	**Constraints**	**L0-default**
`driver_ac_maintenance_mode_requires_full_elec`	Specifies to what degree the driver A/C function, if the vehicle in question is equipped with such, while in maintenance mode, depends on the state of electrics. 0 / “false” means that the driver A/C will switch off completely unless the ignition key is at position 1 (“electrics on” / “ACC”) or greater. 1 / “true” means that the driver A/C will switch off completely unless the ignition key is at position 2 (“pre-ignition” / “ON”) or greater.	Boolean	1
`passenger_ac_maintenance_mode_requires_full_elec`	Specifies to what degree the passenger A/C function, if the vehicle in question is equipped with such, while in maintenance mode, depends on the state of electrics. 0 / “false” means that the passenger A/C will switch off completely unless the ignition key is at position 1 (“electrics on” / “ACC”) or greater. 1 / “true” means that the passenger A/C will switch off completely unless the ignition key is at position 2 (“pre-ignition” / “ON”) or greater.	Boolean	1

B.7. Miscellaneous / visual

Subsystem name prefix	`visual`		
L1 const-file	`visual_constfile.txt`		
Setting name suffix	**Description**	**Constraints**	**L0-default**
`has_static_registration_identifiers`	Specifies whether the identifiers (license plate registration number and operator-specific number), generated dynamically by OMSI for each spawned vehicle, are to be displayed (0) or not (1). This setting is intended for use with repaints, such as the “native” CSB ones, that incorporate those identifiers into the design (due to there principally being a “one-to-one” relationship between .bus- and .cti-file).	Boolean	0

C. AI lighting subsystem diagnostic mode

Purpose

The AI lighting subsystem is a rather complex piece of AI logic, with over 10 independent, and tens more of dependent variables contributing to the behavior of each vehicle. This makes it hard to debug by mere observation, because, as they say, statistics tend to defy intuition and common sense. For this reason the AI lighting script includes a testing facility – a kind of *simulation within the simulation* –, through which multiple AI lighting "profiles" can be obtained via a single vehicle at short intervals, for subsequent semi-automated analysis. We have chosen to expose this functionality to users in the hope that it will provide them with insight in those cases where casual observation won't; for example, when witnessing five vehicles in a row using fog lights during the day, one may be tempted to believe that there is a flaw with the implementation – an assumption which the metrics produced by this utility will hopefully prove wrong.

In summary, aided by the diagnostic mode, one may:

- Deduce how, on average, vehicles behave under the current environmental conditions.
- Verify that the specified constraints hold.
- Verify that less frequent cases specified, that are hard to witness via manual testing, actually occur.
- Infer, in terms of logical consistency, the correctness of the specification itself.

Usage

1. Start OMSI.
2. Map the `lights_ai_test_toggle` trigger to some key (combination) of your input device.
3. Select a map and set the new situation's time and/or entry point to values that are *unlikely* to cause the game to load any buses, initially; e.g. a time past midnight and/or an entry point where scheduled AI traffic is infrequent.
4. Load the map.
5. Adjust the weather and time to your liking.
6. Change the time, if necessary, such that an AI bus gets spawned in your proximity. It is important that an (initially) AI-controlled bus be used.
7. Take over control of the vehicle while it is still moving, i.e., before it reaches any scheduled stop, otherwise observations may be inaccurate.
8. If the weather is wet, fully stop the vehicle, otherwise observations may be inaccurate.
9. Activate the testing mode via `lights_ai_test_toggle`. A sound should be emitted to confirm activation.
10. Wait until you hear that same sound again. A full testing round of generating and processing 10,000 lighting profiles typically takes 5~10 seconds.
11. Quick-save (Ctrl+S) the game.
12. Open `<OMSI2>\maps\<map_you_were_on>\lastsn.osn`. Copy everything between `lights_ai_test_status` and the line following `lights_ai_test_ctx_ext_off_and_int_on_percent` to a new file. Ensure that a) `lights_ai_test_status` has the value 0, and b) `lights_ai_test_ctx_total_configs` has a value greater than 0; otherwise testing likely never commenced in the first place.

13. Study the values of the variables recorded.

Core metrics reference

`lights_ai_test_ctx_ext_off_percent[[_without]_drl]`
Percentage (x/10000) of vehicles (equipped, or not equipped with DRL, or both combined) not using any exterior light source, under the current conditions.

`lights_ai_test_ctx_standstill_percent[[_without]_drl]`
Percentage of vehicles using standstill/position lights, under the current conditions.

`lights_ai_test_ctx_standstill_and_head_fog_lights_percent[[_without]_drl]`
Percentage of vehicles using standstill/position lights *and* head fog lights, under the current conditions.

`lights_ai_test_ctx_headlights_percent[[_without]_drl]`
Percentage of vehicles using headlights, under the current conditions.

`lights_ai_test_ctx_headlights_and_head_fog_lights_percent[[_without]_drl]`
Percentage of vehicles using headlights *and* head fog lights, under the current conditions.

`lights_ai_test_ctx_headlights_and_full_fog_lights_percent_without_drl`
Percentage of vehicles using headlights *and* full (head and tail) fog lights, under the current conditions.

`lights_ai_test_ctx_ext_on_for_seeing_percent[[_without]_drl]`
Percentage of vehicles using any exterior light source for the purpose of seeing, under the current conditions.

`lights_ai_test_ctx_ext_on_for_seeing_reduced_percent[[_without]_drl]`
Percentage of vehicles using any exterior light source for the purpose of reduced seeing, under the current conditions.

`lights_ai_test_ctx_ext_on_for_seeing_adverse_percent[[_without]_drl]`
Percentage of vehicles using any exterior light source for the purpose of adverse seeing, under the current conditions.

`lights_ai_test_ctx_ext_on_for_being_seen_percent[[_without]_drl]`
Percentage of vehicles using any exterior light source for the purpose of being seen, under the current conditions.

`lights_ai_test_ctx_int_off_percent`
Percentage of vehicles not using any interior light source, under the current conditions.

`lights_ai_test_ctx_int_dimmed_percent`
Percentage of vehicles using reduced/dimmed interior lighting, under the current conditions.

`lights_ai_test_ctx_int_full_percent`
Percentage of vehicles using full interior lighting, under the current conditions.

Tips

- Percentages are expressed in the interval [0, 1], 0 = 0%, 1 = 100%.

- Both the time and weather conditions are set in stone throughout test execution. Ensure they are correct *before* initiating the test mode.
- Using a "freshly-spawned" AI bus for each set of to-be-tested conditions is recommended.
- Variables with a name prefixed "_ctx_" refer to metrics that depend on the testing context (time, weather, etc.). These should be the primary focus. The remainder of the `lights_ai_test_xxx` variables are mainly destined for debugging.
- The game *will* significantly lag while testing is in progress. This is normal – more or less the whole AI lighting script is run 100x/frame, to minimize the overall time each full testing round takes to complete.
- There is no guarantee of 100% accuracy in recorded metrics. Rounding errors after the first few decimal places are commonplace in OMSI.
- Avoid testing under conditions that are *both* windy *and* rainy/snowy; the error margin, due to precipitation dependence on vehicle orientation versus wind direction, will greatly increase (the script cannot currently reliably normalize precipitation under such conditions).
- Avoid testing under conditions of optically dense but *scattered* clouds (such as the standard “Cumulus 2” and “Cumulus 3” ones; the error margin, due to brightness dependence on vehicle location, will greatly increase. Instead use the provided “Haze” and “Dust” types of clouds, that induce a more uniform brightness reduction.

D. Known issues

This section lists a subset of all bugs that are yet to be resolved and are of considerable severity. For a full enumeration of open issues – both problems and thoughts on future work – the reader may refer to the project's issue tracker on GitHub.

OMSI crashes upon quitting and situation change

When playing on a map where instances of the enhanced vehicles were spawned, either due to the AI-list or directly by the user, the OMSI process will refuse to gracefully exit upon attempting to quit it, producing an "app-hang" or "app-crash" report. Reloading the same situation, or loading a different one, will too cause OMSI to become unresponsive.

As a workaround, the user may quit OMSI via Windows' task manager.

The cause of this bug remains unknown.

OMSI raises error when enhanced vehicle assigned to *unscheduled* AI

The game may raise a "Zugriffsverletzung" ("access violation") error when an enhanced vehicle not currently on a schedule gets assigned to the AI.

This bug is also present in other Citaro variants, both plain vanilla and CSB adaptations. This is suspected to be the response of the game to a script attempting to query the timetable without first checking whether the vehicle indeed has been assigned one. This modification's scripts have all been double-checked to ensure they are not the cause. Preexisting unmodified scripts making use of such functionality, like predominantly the scripts driving destination display, ticket printer, and IBIS devices, are thus likely candidates.

Due to the bug hardly ever manifesting – that is, unless the user provokes it, by e.g. registering enhanced vehicles as unscheduled AI traffic with a corresponding AI-group, or manually, in-game, allowing the AI to take over a vehicle without an assigned schedule – we are disinclined to invest time into its resolution, as things currently stand. Furthermore, due to some of the matrix scripts relied on by the CSB vehicles being copyrighted, we would, out of redistribution restrictions, not be (conveniently) capable of patching them, if they turned out being responsible.

Humans crossing the vehicle's path at a stop may prevent it from departing

When, just before its departure from a scheduled stop, a vehicle's path gets crossed by humans (i.e., walking across the street right in front of the still stopped vehicle), the humans and the vehicle may enter a situation, where both parties block each other, neither willing to take the first step to resolve the "deadlock".

There might be a link between this bug and our enhanced departure work-flow – vehicles telling OMSI that they departed before actually having done so, primarily to circumvent the stop overlapping effect. Much experimentation and testing is needed for a definitive “diagnosis”.

As a workaround the user may, in F4 camera mode, distance themselves from the affected vehicle by one or more tiles, depending on their configured number of loaded tiles. OMSI should then either de-/re-spawn the blocked vehicle, and/or de-spawn the troublesome humans.

The AI cannot recover on its own from (severe) crashes with user-vehicles

When a user - AI crash causes significant damage (electrics and/or engine and/or transmission malfunctions) to the AI vehicle involved, the latter will stay damaged, “refusing” to behave normally, even after the user has “called the police”. The damaged AI vehicle has thus the potential of blocking traffic anywhere it gets subsequently placed by the game.

This is partly bug, partly feature. It is a feature because the original intent was for the AI to “deny” taking over severely-damaged user vehicles, until repaired by the user themselves (under standard scripts the AI vehicle would magically appear to drive on, even without an operable engine, which was deemed plain ugly). It is a bug because we did not anticipate this particular variation – the user’s vehicle causing damage to another vehicle that is already under the AI’s control), until beta-testing revealed that OMSI does not automatically trigger the repair function for crashed-into AI vehicles.

A fix for this issue will likely be made available in a future release.

User level workarounds include:

- Disabling collisions; or
- taking over control of damaged AI vehicles and de-spawning them; or
- taking over control of damaged AI vehicles and repairing them.

OMSI may raise error when vehicle repair gets requested

When the user triggers the repair function of the native game menu, OMSI may respond with an “Integer Überlauf” (“integer overflow”) error. Repairs are still carried out. The same error may be issued anew when subsequently attempting to terminate the game or reload / change situation, likely leading to an OMSI process “hang”.

The bug manifests when pending repairs require (according to OMSI’s idiosyncratic nature) “too much” time – roughly above 200 minutes. Since there appears to be an internal OMSI issue involved here, little can be done – except limiting repair duration, which would not be a true solution either…

No support for saved situation reload

Reloading a previously saved situation that involves vehicles of this modification, is a use case that is currently not formally supported.

“Not formally” means that the scripts are at present not designed with the function of saved situation reload in mind, and no testing has been conducted to ensure they work properly when that function is employed. Therefore, use of this function is discouraged, and no support will be offered to users experiencing problems in the context of reloaded vehicles.

Further background is available in this issue tracker entry.

While not being able to offer any “hard” guarantee, adhering to the following guidelines may help alleviate some (but not all) issues that are suspected to manifest upon saved situation reload:

- Only saving when the vehicle is stationary.
- Ideally, only saving when the engine is not running, and transmission is in neutral.
- Ideally, only saving while not driving on a schedule.
- Not changing the situation’s date and/or time while in the main game menu, before reloading the situation (changing the time and/or implicitly the date, after reload has completed, is fine, however).

E. Changelog

E.1. Changes in 1.0.0 (06/2018)

AI-specific functional changes

1. Introduced preference-based AI usage of interior and exterior lighting to the CVAG 72 vehicle – a two-door E3 Citaro – of the 1.30 CSB modification package.
2. Added simplified interior illumination effects to unfocused vehicles, to prevent them from appearing unrealistically dark at night, as used to be the case.
3. AI passengers no longer complain about the vehicle's interior being too dark under lighting level 1.
4. When headlights are active, manual high beam toggling at short intervals by the user now too gets interpreted as "flashing" by other AI traffic.
5. Lighting switch triggers are now disabled while vehicle is in AI mode, to reduce complexity of AI logic.

Other functional changes

6. Under interior lighting level 1, when the engine is on, the third row of passenger tubes now activates as well.
7. Fog lights now have independent triggers from the ones driving standstill light and headlight activation and deactivation. This allows usage of just standstill lights alongside fog lights, which was previously not supported.
8. Added missing money illumination.
9. Standstill and position lights now remain active even when electrics have been completely disabled, provided the battery can cope. Likewise, high beam flashing is always allowed, regardless of engine's and electrics' state.
10. Revised fog light activation conditions: a) At least standstill lights must be on, and b) either the engine must be running, or the fog light mode must be maximum (both head (if installed) and tail fog lights).

Non-functional changes

11. Formalized the AI lighting requirements in a specification-style document.
12. Relocated most lighting switch related triggers and logic to cockpit script, for better organization and coupling reduction. Removed obsolete "ober-"/"unterdeck" logic; retained legacy trigger names for user convenience, for the time being.

E.2. Changes in 1.0.1 (06-07/2018)

Non-functional changes

13. Implemented AI lighting subsystem diagnostic mode for debugging / verification convenience.

E.3. Changes in 1.1.0 (07-08/2018)

AI-specific functional changes

14. Made a few more objects visible in unfocused vehicles, including the steering wheel, the driver's seat, and the dashboard's door buttons. Furthermore, in unfocused vehicles, a simplified texture now gets applied to the VDV display, which was previously completely dark.
15. Added pseudo-illumination effect to more objects in unfocused vehicles, including the cockpit ceiling, driver cubicle's door, dashboard, steering wheel, ticket validator, interior door surfaces, and seats. Furthermore applied illumination effect distinction based on the object's location – for cockpit objects the driver's light must be active, whereas for other objects any passenger light source being active suffices for its illumination in unfocused mode.
16. User→ AI signaling improvements in user-controlled vehicles:
    - For standstill lights to be perceived as active by other AI vehicles, a) at least one source (standstill, marker, head- or fog light) at the vehicles head needs to be active, and b) at least one marker light on both sides needs to be active, and c) at least one light (tail, marker, or fog light) at the vehicle's tail needs to be active.
    - For indicators to be perceived as active by other AI vehicles, accordingly per side, at least a) the head or side, and b) one tail indicator light needs to be active.
    - For cabin lights to be perceived as active by AI passengers, at least one interior fluorescent light tube must be fully functional (tubes in a partially defunct / "worn-out" state don't count).

Other functional changes

17. Enhanced interior light sources and states thereof to be more realistic. There are now 6 individually operable passenger cabin fluorescent light tubes, as opposed to the original number of 8 tubes controllable in 2 groups of 4. As a consequence, most light-source-to-mesh and to-passenger-seat mappings had to be revised as well, as had the night-maps of the perceived light emitters themselves. Given this opportunity, the speakers' (texture) positions was brought closer to reality as well.
18. Every exterior and interior light source (excluding dashboard indicators) now consumes power individually (integrated with electrical subsystem), and can individually become defunct (integrated with malfunction subsystem and game-native repair function), either due to collision-induced damage (exterior lights only), or "natural" end-of-life. The likelihood of either occurring depends on a) the type of the source (e.g. LED lasting longer than fluorescent, in turn lasting longer than halogen), b) the significance of the source in terms of safety and comfort (e.g. burnt out door spotlights tending to be replaced less frequently than the driver's light), c) the location of the source (e.g. orange markers directly at the front being more

susceptible to micro-collisions than markers in the middle), d) the user's maintenance setting (randomized for buses initially spawned in AI mode), and e) the vehicles age (present year - year of manufacture).

Furthermore, when the battery is critically low and the engine has been switched off, circuit level light source groups of a lesser significance may be preemptively disabled automatically to conserve energy.

Lastly, fluorescent tubes may enter a partially defunct / "dimmed" state, sometimes long before burning out. In that state they have a different night-map texture and do not actually illuminate their surroundings (out of OMSI limitations).

19. Introduced beam effects and, in some cases, new night-map textures, to vast majority of exterior light sources not already equipped with such.
20. Door spotlights now actualize when electrics are off (they previously would remain in whatever state they used to be in, until electrics were re-enabled). The same applies to the case where door air pressure is insufficient for door operation, that is, door spotlights are unaffected by blinking dashboard door buttons. Lastly, door spotlights now instantly go off irrespective of the post-door-closure delay, when exterior lighting has ceased being active. Previously, if, just after a door had closed, but prior to its spotlights' deactivation delay expiry, exterior lighting got toggled off and on again, the spotlights would too go off and on again, as opposed to going off and remaining off.
21. The state of the driver's light now instantly synchronizes to the state of the front door's spotlights. Previously there used to be a slight synchronization delay.
22. Light source lifetime now gets decreased continuously post-spawn once again, as happens in the standard OMSI vehicles (in the Citaro the pertinent logic had been removed for some reason, thus lights would never burn out post-spawn).
23. On saved situation reload, i.e., on saved vehicle re-spawn, light source lifetime no longer gets re-initialized.
24. The VDV now reports damage of any exterior light source, as opposed to solely indicator, headlight, and high beam bulbs, as it used to.
25. The VDV icons reflecting percentage of air pressure and fuel availability no longer get "stuck" rendered on display when electrics get abruptly shut down while the display is still undergoing initialization.
26. The VDV now only reports "consumption warning" when light / indicator sources are actually active, i.e., when they actually consume power, as opposed to just being switched on dashboard-wise; for example, if interior lighting is active, yet all passenger cabin tubes have become defunct, no warning will be emitted. Additionally, the "defunct exterior lights warning" is now driven solely by exterior light sources (it would previously also account for the driver's light, which seemed unrealistic, given the displayed message).
27. The collision "breaking glass" sound now only gets triggered when an exterior light source close to the vehicle's head becomes defunct due to collision impact. The sound volume depends on how many sources have broken.
28. Brightened up, in terms of material properties, certain E3-specific objects (or vanilla objects with different E3-specific textures) that had stark contrast, in terms of ambient light absorption, from their surroundings. The driver cubicle's door is a notable example.
29. Plugged a small gap behind the driver cubicle, at the point where it meets the wall/ceiling.
30. Added a level of indirection between light source state (voltage and operational status), and illumination effects (night-maps, beam effects, etc.). This allows for more granular control in general, and in particular prevents night-maps from being disabled prematurely, due to a voltage

below 0.5 being rounded down to 0 at the model.cfg level. Furthermore, spot-select now properly reflects light source state, not circuit state (e.g. previously the environmental illumination effect of the high beam would erroneously be applied at times when the sources themselves were off).

31. Door spotlights now emit actual light (i.e. implemented as dynamic interior-lights, not just light-maps). Interior door surfaces and door handles, as well as nearby AI passengers, are now illuminated by them.
32. Interior lights coupled to doors (driver's light, spot lights, and front-most pair of interior fluorescent light tubes) are no longer affected by door problems (e.g. insufficient pressure). Dashboard door buttons still blink in response to such problems.
33. The passenger cabin's destination display's "STOP" indicator is now always illuminated when active, independent of exterior lighting status (as long as there is sufficient power availability).
34. Certain IBIS / ticket printer screens, such as that of the "departure clearance" overlay, are no longer erroneously rendered when electrics are disabled or unavailable.
35. VDV (dash lights and display), IBIS / ticket printer (display), and passenger information display are all assumed to be connected to an independent battery with infinite capacity, and hence no longer "flicker" when voltage applied by main battery gets critically low (typically manifesting during "cold-start"). This oversimplification was introduced to account for the fact that those devices would otherwise need to properly re-initialize / "reboot" after power loss, rather than instantly becoming operational again, which in turn was deemed to costly to properly implement at the time (may be addressed in a future version).
36. model.cfg interior-light declarations moved from "shadow" to "wagenkasten" mesh, to prevent lights' height to vary depending on shadow's height, which in turn appears to depend on the underlying terrain the vehicle is on. Previously, that fact would induce curious light "flickering" or complete illumination effect invalidation when e.g. driving down a slope or over a bumpy surface.
37. Upgraded VDV subsystem to Morphi's v. 4.4.

Non-functional changes

38. The lighting subsystem now supports DRL (also integrated with spot-select), configurable via .cti-variable, for potential use by future CSB vehicle integrations. Primary activation conditions: a) running engine, and b) exterior lighting (not accounting for high beam flashing) being disabled.
39. The lighting subsystem now supports an optional additional pair of a tail fog light on the right and a reverse light on the left, configurable via .cti-variable, for potential use by future CSB vehicle integrations.
40. Removed seemingly duplicate "wagenkasten(_orlf)" entries from model.cfg.
41. Added inline documentation to lights.osc and lights_ai.osc.

E.4. Changes in 1.2.0 (08/2018)

AI-specific functional changes

42. Introduced enhanced AI usage of doors, wing locking, and the stop brake. Brake pedal released by AI when stop brake or handbrake engaged. When stop brake is to be released by AI, throttle

pedal gets mildly “tapped” first, then briefly released, and finally firmly pushed in order to actually accelerate.

43. Driver cubicle’s door position affects front door’s opening status perceived by AI passengers; i.e., when the cubicle’s door is even slightly open, passengers won’t attempt to board via the front door and thus appear to “walk through” the cubicle’s door intersecting with their path. Out of OMSI limitations this still doesn’t prevent the same problem from arising when the cubicle’s door gets opened after passengers already have crossed the entry boundary but have not yet walked past the driver’s cubicle.
44. To facilitate subsequent AI enhancements, the user’s ability to trigger any cockpit, VDV, IBIS / ticket printer, and cash desk switch, as well as emergency door opening valves, the ramp, and the driver cubicle’s door, when the vehicle is AI-controlled, has been revoked. Sole exceptions are passenger windows and stop request buttons.

Other functional changes

45. Hatches are now only operable when electrics are on; they also no longer adjust instantly.
46. Various dashboard switch indicator lights, used for signaling corresponding function state, no longer remain active when electrics are off, including those of the “school driver” indicator mode’s switch, and the side panel’s heating / cooling buttons and turn-switches.
47. The auxiliary heating button now emits a sound when pushed / released.
48. Integrated Morphi’s v. 4.4 E3-ZF sound set (previously Voith sounds were in use).
49. Integrated “UCHill” (alternative heating / cooling subsystem), featuring non-instant adjustment of heating / cooling functions, more robust humidity management, a greenhouse-like effect, and a reduced / “maintenance” / fan-only mode for the driver and passenger A/C functions.
50. Fine-tuned min/max volume of exterior sounds on the inside, as well as how much individual open-able surfaces (doors, windows, hatches) contribute.
51. Direction indicator improvements:
    - Activation in “school driver” mode is no longer delayed.
    - When indicators are already active in simple “left-or-right” mode, triggering the “hazard” or “school driver” mode causes the interval timer to reset and hence indicator lights on both sides to activate simultaneously at that exact time.
    - When indicators are already active in “hazard” or “school driver” mode, but indicator lever also set, disabling the overlapping / overridden / simple “left-or-right” mode no longer affects the timing of the effective mode.
    - All relevant dashboard indicators are now exactly synchronized to the interval timer and therefore to the exterior indicator lights as well.
    - Inverting (left $\leftrightarrow$ right) the indicator lever triggers two sounds – both “off” and “on”. Activating (off $\rightarrow$ left/right) or deactivating (left/right $\rightarrow$ off) the indicator lever triggers a single sound (“on”, “off”, respectively). There were previously multiple overlapping triggers in the cockpit script, which in certain cases erroneously emitted too many or too few sound (or none at all). These triggers were all cleaned up and merged into just three – “set-left”, “set-right”, “set-off”.
    - When electrics are off, indicators only blink when in “hazard” or “school driver” mode. Indicator light source group output is cut off in simple “left-or-right” mode.
52. “Hand-brake-not-engaged-while-engine-off” buzzer no longer incorrectly triggered when electrics on but circuit damaged due to collision; the same holds for the “schnaufen” and “ambience” sounds.
53. The main sun blind is now fully retractable.
54. Less flexible but more deterministic and fault-tolerant door wing locking logic:

- Locking mode (“rear wing” / “front wing” / “none”) can now only be adjusted when both wings are closed. While the switch may still be flipped at all times, the effective mode will not change unless that precondition has first been met.
- When switch flipped while doors are in the process of adjusting, individual wings can no longer become stuck open or closed; nor can one wing open and the other close, as the result of a single “keystroke” (button pushing).
- VDV correctly depicts the effective state of doors (both in terms of opening and of locking status), regardless of switch state; e.g., if only front wing open, yet wing lock switch set to “front wing” or “none”, VDV will still display just the front wing as open, and the rear one as closed (and potentially locked, depending on past switch state that became effective).

55. The stop brake does no longer automatically get re-engaged when door closure gets triggered.
56. One can no longer “trick” the stop brake into engagement and instant disengagement thereafter, when triggering door opening prior to having stopped, and proceeding to accelerate anew immediately thereafter.
57. Animated indicator lever wiper wash mode.

Non-functional changes

58. Moved door wing locking and “school driver” indicator mode switch triggers to cockpit script.
59. Removed a significant amount of leftover cockpit script triggers and macros used by long-obsolete MAN functions, as well as functions specific to non-German-typical Citaros (such as sliding passenger windows encountered in eastern EU and Russian variants; stand-alone fog light switches; duplicate heating and cooling controls). Likewise removed unnecessary “clutter” from door script (support for 3-/4-D variants and automatic doors). The rationale for these changes was simplification and consolidation in order to improve testability. Some functions may be re-introduced on an as-needed basis in later versions, to accommodate for other variants’ needs.
60. Consolidated door opening status into a single block of logic, intended for consistent reuse by other subsystems for:
    - Evaluation of opening / closing preconditions.
    - Dashboard door button illumination.
    - Door spotlight activation.
    - VDV door opening status icon rendering.

    Prior, more or less the same logic would be repeated whenever needed. Also, dashboard door button illumination has been synchronized with VDV door status icon rendering status.
61. Precautionary measures for AI-controlled vehicles:
    - Door adjustment does not induce decline of air pressure.
    - Fuel consumption is zero (in the sense that the vehicle retains the exact amount of fuel it was spawned with, for as long as it remains under AI control).
    - Engine ignition always is always possible and in a timely fashion, even if battery old or depleted.

E.5. Changes in 1.3.0 (09/2018)

AI-specific functional changes

62. Introduced AI usage of sun blinds.
63. Introduced AI usage of the driver's window.
64. Introduced AI usage of heating / cooling functions. For now support is limited to:
    - The driver A/C (temperature, fan speed, air re-circulation)
    - Passenger A/C
    - Cabin heaters
65. Introduced AI usage of hatches.
66. Introduced asynchronous, randomly-delayed sequence of engine ignition / switching off, switching between 'D' and 'N' transmission modes, and handbrake (dis-)engagement, upon AI arrival at / departure from prolonged stops.
67. Enhanced AI wiper usage.

Other functional changes

68. Windows are now illuminated by primary passenger cabin lights.

E.6. Changes in 1.3.1 (10/2018)

AI-specific functional changes

69. AI can now detect when next stop is located directly adjacent to current stop (two bus stop cubes being close enough such that their docking distances overlap) and refrain from *visibly* departing (closing doors, etc.). This is a typical occurrence at terminal / initial stop pairs, such as the terminus at "U-Bhf. Ruhleben" on the M+R Spandau map, where buses with standard AI logic will *seem as if they were to depart*, only to *instantly seem to arrive* again (re-open doors, etc.), without having moved a single centimeter in between stop cubes.
70. AI can now detect contextual anomalies, such as the user having changed the time, or the vehicle itself having been "hibernated" (typically due to the user leaving its tile) by the game, and "revived" at some arbitrarily later point in time, likely after having been re-spawned to a different location, according to its schedule. Awareness of the fact empowers AI vehicles with the ability of instantly performing any long-delayed actions already scheduled before "hibernation" (such as altering the state of lighting) under such conditions, so that the observing user can't (easily) tell they were in fact "frozen in time", as well as minimize downtime (e.g. blocking a busy junction for a minute or two while awaiting their usual departure routine to complete), while otherwise still behaving naturally.
71. AI can now detect when a departure is in reality a *parking maneuver*, and behave appropriately (by, if needed, turning on the engine sooner, and occasionally skipping closure of the front door, if applicable). This occurs upon arrival at stops with a long docking distance, where another vehicle (typically another AI bus) happens to block their path, but still departs prior to the blocked vehicle's departure time having arrived. The game then instructs the blocked vehicle to depart and stop again, at the same stop, but at the proper location, bus-stop-cube-wise.

72. Reduced frequency of blocking occurrences of front door's first wing in AI mode. This is only a temp-fix to account for the fact that the AI is not yet "smart" enough to detect blocked wings and resolve the issue like a human user would (by repeatedly re-opening and re-closing them, or employing the emergency opening valves).
73. During AI takeover of a previously user-controlled vehicle, unsupported AI functions (e.g., retrder, auxheat) are simply disabled, to avoid implicitly influencing AI logic, or merely appearing erratic / dumb. In the same spirit, the AI is now capable of (somewhat) gracefully handling the following less trivial situations:
    - When an emergency opening valve has been left active, the AI will reset it and properly close the corresponding door afterwards.
    - When the rear door wheelchair ramp has been left unfolded, the AI will realistically re-open the rear door if need be, then realistically drag the ramp back up, and close the rear door once again.
    - When the driver cubicle's door has been left open, the AI will drag it back to its fully closed position.
    - If transmission has been left in 'R' mode, the AI will switch to 'N' and then 'D'. If additionally the engine was left off, the AI will reignite it in between, i.e., before re-engaging 'D'.
    - If the IBIS / ticket printer device display mode was changed, the AI will reset it. The same applies with regards to the VDV display mode.
    - If the IBIS / ticket printer device had previously issued a "departure clearance" message, and the user did not acknowledge it, then the AI will.

    Note that some of these cases are still not realistically handled in the extra-special case where the user assigns their vehicle to the AI *while moving* – more work will be required to address this, possibly in a future version.
74. During AI takeover of a previously user-controlled vehicle, which is severely damaged (broken engine, transmission, or electrics), the AI will enter a "failsafe" state where it will plainly refuse to magically operate the vehicle, instead opting to do just nothing until the user has repaired it (via the native menu function). The only actions the AI will then be willing to undertake are turning off the engine, if running; switching transmission to 'N', if engaged; engaging the handbrake, if disengaged; and activating the "hazard" indicator mode, if electrics are still functional.

E.7. Changes in 1.3.2 (10/2018)

AI-specific functional changes

75. AI can now detect and disregard a stale stop request issued prior to "hibernation". Previously, "revived" AI vehicles would open the rear door upon their next scheduled stop arrival, due to "remembering" that a stop request had been issued before "hibernation" – in spite of the fact that those passengers that were willing to disembark way back when are no longer present (OMSI tends to de-spawn boarded passengers from "hibernated" AI vehicles).
76. AI now closes doors as soon as possible under (subjectively defined) cold or hot conditions, when the A/C is active. Furthermore, for added realism, the AI will try to predict, based on its schedule (albeit not always successfully) whether a stop is only going to last for a few seconds,

and, if so, closing doors asynchronously once each of them is no longer needed for boarding or disembarking, in anticipation of a speedy departure.

E.8. Changes in 1.4.0 (10/2018)

AI-specific functional changes

77. Enhanced AI usage of direction indicators.
78. Introduced separate, more lightweight sound configuration file for unfocused vehicles.

E.9. Changes in 1.5.0 (11-12/2018)

AI-specific functional changes

79. Introduced AI usage of folding passenger windows. Both the perspective of the AI driver, as well as the perspectives of any AI passengers seated close to each window, are simulated.
80. AI buses can now spawn dirty, as well as get dirtier in the process. Also, washing the vehicle now takes a bit longer.
81. During parking maneuvers, the AI no longer:
    - Resets / closes the driver window, if lowered at arrival and hot outside.
    - Resets / activates door wing lock, if disabled at arrival and hot outside.
    - Reactivates cabin lights, if temporarily switched off at arrival.

Other functional changes

82. Incorporated Morphi's v. 4.4 changes to engine, brake, and transmission scripts.
83. Folding passenger windows can now be locked (by user or AI driver, depending on user vs. AI mode), as well as completely disabled per .cti. When locked, AI passengers will be unable to open or close them. When disabled, neither the user themselves, nor the AI driver, nor the AI passengers will be able to open or close them.
84. Several key objects were re-ordered at the model.cfg level in order to attain better "layering" of transparent surfaces. Previously e.g. the black surface in the rear of the driver cubicle used to render "atop" the exterior windows, producing not so-aesthetically-pleasing results, particularly in conjunction with repaints (such as the CVAG 72's) covering transparent window surfaces. These changes also enable certain parts of the exterior to once again become dirty, which they previously would not.
85. The direction indicator "hazard" mode switch can now blink, if configured to do so via .cti.
86. Wipers and direction indicators in regular "left-or-right" mode now require that the ignition key be set to "pre-ignition" / "ON", as opposed to merely *not being "OFF"*. Furthermore, headlights, head fog lights, and primary cabin lights now require that either the engine be running, or that the ignition key be at least at position 2 ("electrics on" / "ACC"). Previously they strictly depended on just the engine running.
87. The "NOT-AUS" switch can now be used to override the coupling (automatic engagement upon opening) between the rear (and front, if configured) door and the stop brake. A warning

message has been added for the VDV to display when the switch is active. Furthermore, the ability to kneel when the switch is active has been removed (the vehicle will automatically kneel up in the event of the switch being triggered while already knelt down).

Non-functional changes

88. Restructured overall script / macro execution order, in an attempt to further reduce coupling and make the processing "pipeline" less prone to logical errors, where a certain subsystem would expect that a task occur after it, when in fact it occurs (due to the execution order) before it but during the *next frame*.
89. Distribution archive changes:
    - Added independent .bus and model, passenger-cabin and sound configs.
    - Added independent directory structure for scripts, modified objects, and textures thereof.
    - Authored trigger / .cti-variable reference.
    - Compiled this very changelog.
    - Authored AI lighting diagnostic mode guide; included new cloud textures which might be useful for testing purposes.

E.10. Changes in 1.5.1 (01-03/2019)

AI-specific functional changes

90. Vehicles spawned "in-parallel", e.g. at situation load time, frequently seemed to exhibit *too many* similarities, in terms of AI behavior and other vehicle traits. The initialization flow of several scripts has been adjusted in an attempt to circumvent this, although we fear there might be an underlying OMSI issue there (worst case: random number generation, on which the vast majority of AI logic heavily depends, producing the same output across vehicles when requested at the same time or game frame).
91. AI "failsafe" (see changelog item 74) now also triggers when tank without fuel, or battery depleted (but otherwise operable).
92. AI passengers' temperature perception is now driven by a combination of environmental and cabin temperature, as opposed to solely the latter. This is of significance when the vehicle is user-controlled: The cabin temperature may then rise several degrees above the environmental temperature due to insolation (greenhouse-like effect), to which passengers may then respond by being more willing to open windows, even when it is not really that warm / comfortable outside.

    Note that this does not really apply to AI-controlled vehicles. First and foremost, window adjustment by passengers only occurs in focused vehicles (which is not usually the case for AI-controlled vehicles, since the user tends to have their own vehicle in focus), due to OMSI not communicating seat occupation status to scripts in unfocused vehicles. Second, in AI mode OMSI alters the cabin temperature such that it doesn't depart from its environmental counterpart too much, which then translates to a much more constrained insolation-induced temperature increase, and hence poses less of a motivation on passengers to open windows, unless the environment itself starts getting uncomfortably warm.

Additionally it should be noted that the driver's temperature perception remains entirely driven by the environment for the time being, primarily out of reasons of complexity; that may change in a future version.

93. Unconditional ("eager") door opening at prolonged stops now varies per stop, based on dynamically-evaluated preference, in addition to environmental perception; previously that preference was static, as long as the environmental conditions remained unchanged.
94. Accounted for an AI stop work-flow corner case, where door closure would not successfully occur when vehicle re-spawned just about the time its doors were beginning to open (due to stop arrival the instant before), causing the vehicle to either remain stuck in perpetuity, or erroneously drive on with an open front door.
95. Fixed a bug where an AI vehicle would, during a parking maneuver departure sequence just shortly after its last arrival, magically manage to ignite the engine despite not having had enough time to switch its transmission to 'N' first.
96. AI direction indicator inversion during scheduled stop departure (e.g. "left" if previously indicating "right") now occurs closer to the actual time of departure than it used to (previously AI vehicles had the tendency to begin indicating their departure way too early).
97. The AI can now gracefully handle the situation where it needs to adjust the state of the driver window or the hatches, yet the state of its electrics do not permit doing so (because the AI reduced their operational mode as a result of turning off the engine upon prolonged stop arrival); for example when the weather gets warmer while the vehicle is stopped. The AI will now, if need be (i.e., if the intended operation, depending on the corresponding .cti-var, is constrained by electrics), temporarily increase the mode of its electrics, carry out all apposite function state adjustments, and, once complete, reduce electrics once again.
98. The AI can now manually and momentarily toggle wipers on and off again, under dry conditions, when wiper blades "stuck" at a non-zero angle (due to having previously been deactivated prematurely, because of electrics mode reduction in response to prolonged stop arrival).
99. Minor changes to AI "dirtiness" algorithm, such that overall fewer AI vehicles become significantly dirty under good weather conditions, yet some still do.
100. Addressed a bug preventing the wiper turn-switch's trigger from being disabled when the vehicle is under AI control.
101. Addressed a bug where the AI would always fail, i.e., "neglect" to adjust (invert or disable) their direction indicators upon scheduled stop arrival, when previously a parking maneuver departure had occurred. This edge case would only manifest when vehicles would not be instructed to turn their engine off (since they would otherwise usually disable indicators as a consequence, thereby "masking" the problem).
102. Reduced overall probability of vehicles initially spawned in AI mode having defunct light sources.
103. Reworked pass-pos (passenger seat and standing place)→interior-light mappings once again.

Other functional changes

104. The 4 green indicators adjacent to the VDV display, that are not "wired" to any function in particular, now briefly light up during dashboard initialization (electrics "power-up").
105. Wipers no longer get "stuck" temporarily, or de-synchronized, in interval mode, if previously interrupted due to power loss. Likewise, no de-synchronization occurs when wipers get deactivated while being unavailable due to the electrics being unavailable.
106. New .cti-variables for:

- Front door wing locking switch availability (+ AI integration).
- Whether, under a non-running engine, the following functions are allowed / “unlocked” if electrics are a) at least partly on (ignition key ≥ position 1), or b) fully on (ignition key ≥ position 2):
  - Simple “left-or-right” indicator mode.
  - Hatch adjustment.
  - Driver window adjustment.
  - Driver and passenger A/C fan operation (in reduced / “maintenance” mode).
  - Wiper operation.
- Whether, under a non-running engine, cabin lighting level 2 (all tubes on) is allowed / “unlocked” at a) fully-enabled electrics (ignition key ≥ position 2), or b) never.

107. Vehicle age now contributes less to light source malfunction probability; maintenance frequency is what matters most.

108. The 4 buttons on the heating / cooling side panel are now toggle-able: They no longer remain visually pushed-down once mouse or keyboard key(s) have been released. Also corrected triggering behavior (toggle on → release) sound-wise.

109. The roof-mounted A/C unit no longer emits a sound on the outside when solely the driver A/C function is active; either the passenger A/C and/or the humidity management function must be active for the unit to be audible.

110. The fan speed of the passenger A/C function (both functionally as well as sound-wise) now automatically adjusts proportionally to the difference between the current cabin temperature and target (ideally-attainable, minimum for cooling or maximum for heating) temperature.

111. The sound of the driver window sliding upwards or downwards does not get emitted anymore when electrics become disabled while the window switch is still held down. Previously the sound would be emitted perpetually, without the window actually moving, until power supply would be restored.

112. When a) wiper blade(s) at non-zero angle, and then b) electrics reduced or disabled completely, depending on configuration, then c) wipers switched off, then d) upon reactivation of electrics, wipers won’t move. Furthermore, if wipers subsequently reactivated, their angle will increase again, even if during their last, interrupted run it was on the decline (i.e., new wiper phase = min(previous phase, 2 * π - previous phase)).

Non-functional changes

113. Documentation and distribution archive changes:
- Dropped “specification-style” approach for AI functionality – it would take too long to complete, and be too tedious to maintain by a single editor in the long run, as well as too daunting for the average end-user to fight themselves through. Instead “compressed” existing specification on AI lighting into a simpler, non-technical format, and added summary descriptions on the rest of the AI features introduced since. Merged that new document with other supplementary documentation that used to exist as independent documents up to this point.
- Added installation section to new readme.
- Added known issues section to new readme.
- Added credits and legal sections to new readme.
- Added independent repaint subdirectory to distribution archive.

114. Made power availability testing logic more robust across all scripts. Previously code would falsely interpret non-zero voltage applied by the battery as “the battery not being

depleted, therefore providing energy", when in fact, in those cases where the voltage happened to be negative, it should have been interpreted as "the battery actually *being* depleted and *requiring itself* energy to recharge".

115. Added some inline documentation to ai_pre_cockpit.osc.

E.11. Changes in 1.5.2 (03-05/2019)

Other functional changes

116. Backported CSB v. 1.40 changes:
    - CVAG 72 is now known as ETP 3.
    - New repaint (directory).
    - New interior passenger information display.
    - Removal of emission standard stickers.
    - Dashboard hatch switch not available by default anymore (introduced new .cti-variable for those wishing to reinstate it).
    - Small changes to IBIS and door scripts.
    - Button-left VDV control light now "wired" to signal wheelchair entry / exit request.
117. ECAS leveling now only triggers when doors closed and either handbrake or stop brake engaged.
118. Proper synchronization between bottom-left VDV control light and corresponding VDV display wheelchair icon.
119. Added external front door opener (bumper "black dot").
120. The VDV "hazard" indicator mode control light now lights up during initialization as well.
121. Integrated with ETP 11, another two-door E3 Citaro.
122. Auxheat now emits start-up / ignition sound when activated.
123. IBIS / ticket printer and cash desk now pseudo-illuminated in unfocused vehicles as well, when driver's light active.
124. Reinstated `blinker_[left|right]_move` indicator lever triggers. They had been removed (item 51) in the assumption they are duplicates. It turned out they are useful to steering wheel peripheral users.
125. Keyboard users can no longer trigger both left and right direction indicators at once (by simultaneously holding down both, by default, the '7' and '9' num-pad keys); these triggers are now mutually-exclusive, with the one triggered first taking precedence over the latter, until release of the apposite key. Previously both events would be registered, with the ultimate consequence of all indicator lights remaining "stuck" on until release of either of the two conflicting keys.
126. Automatic indicator deactivation due to steering can now be prevented, by keeping the key mapped to the corresponding trigger pushed, thereby virtually "holding the lever in place", preventing it from being reset. Note that for the time being this is only supported for the keyboard-specific triggers since, apparently, the physical levers of (most?) steering wheel peripheral input devices do not act as buttons, in the sense that all their states ("middle", "up", "down") are "stable", and hence cannot accommodate "push-hold-release" logic.

E.12. Changes in 1.5.3 (09-12/2019)

AI-specific functional changes

127. While at a prolonged stop with a non-running engine, if the AI had temporarily increased its electrics mode in order to adjust the driver window or hatches, it will now wait a few more seconds after the electrics-dependent function's adjustment has completed, before reducing its electrics mode again. Previously it would reduce its electrics mode simultaneously to the completion of the apposite function adjustment.
128. Reworked timing of AI engine / transmission / handbrake adjustment sequences at prolonged stop arrival and departure. Previously the three actions – except for, in the case of departure, the engine being reignited before transmission engagement – used to be purely orthogonal to one another, i.e., each adjustment delay was independent of the other two. That led to "exceptional exceptions" – most notably the handbrake being engaged last on arrival, or disengaged first on departure – occurring more frequently than they would in reality. Now all three adjustment delays are based on shared conditional probability, meaning that there is a certain guarantee that, most of the times, the AI will indeed act "by the book", i.e., that it will upon arrival tend to first engage the handbrake, then switch transmission to 'N', and lastly turn off the engine, and that it will upon departure tend to first reignite the engine, then switch transmission back to 'D', and lastly disengage the handbrake.
129. When an AI vehicle gets taken over by the user and subsequently reassigned to AI, if the new AI lighting profile's light source usage preferences differ from those of the previous profile, they will now be applied even when the lighting context has not changed in between the two "AI sessions". Previously different usage preferences would not be reflected in vehicle state, unless the context had changed while the vehicle was under the user's control. This would be particularly evident under good conditions at night, when the context is unambiguously that of "seeing" for all vehicles. Had, for example, the vehicle been spawned with a preference for head fog light usage under "seeing" conditions initially, all subsequent "AI sessions" would appear as if too favoring head fog light usage under those same conditions – even though not true internally.
130. AI lighting actualization upon takeover of a previously user-controlled vehicle now gets randomly delayed by a few seconds, so as to appear consistent with the initial, post-takeover actualization of all the other AI-supported functions, which too are delayed. It would previously always occur at the exact instant of takeover.
131. At prolonged stops where the AI has chosen, either immediately upon arrival, or subsequently because of worsening conditions, to not turn off (or, respectively, to reignite) the engine, it can now successfully both increase and decrease its exterior lights. Previously a bug used to prevent adjustment, and as a consequence exterior lights would statically remain in whatever condition they were previously (prior to arriving or respectively deciding to reignite the engine) in.
132. When the AI chooses, *after* having arrived at a prolonged stop, to reignite the engine because of worsening conditions, and its headlights are already on, it will now manually reduce its exterior lights to just standstill (or standstill + fog lights under adverse conditions, depending on preference) *much sooner* than it would normally. Unlike the case where the AI already knows *at the time of arrival* that it is not going to turn off its engine, which was already taken into consideration previously, this similar yet distinct case was previously not accounted for.

133. Resolved one case of “neurotic” AI braking (rapid braking-throttling-braking cycling), which a script-level bug turned out being the cause for.
134. Made AI aware of additional transmission mode selectors (item 135). The AI simply switches back to ‘D’ if a constrained mode is in use at the time of takeover.

Other functional changes

135. Added support for the additional constrained transmission modes and accompanying dashboard selectors (‘1’/’2’/’3’) thereof:
    - Added cockpit configuration for specifying whether the selectors and triggers shall be available or not.
    - ‘1’ prevents upshift above first speed; ‘2’ above second; ‘3’ above third; all three prevent further upshifting, even if the current speed exceeds the mode’s maximally allowed one.
    - Transmission output torque gets limited above a certain hard limit, depending on engaged gear; it is no longer possible to reach terminal velocity in first, second, or third gear. Added configuration (for the time being not integrated with the reworked configuration scheme, due to being gearbox-specific) for specifying whether the gearbox should enforce an upshift when that limit is exceeded (typically when accelerating downhill) or not.
    - Passive gravity-induced acceleration now also causes torque converter lock-up, allowing the engine braking effect to be felt more strongly (and assist downhill driving) in the lower gears. This also resolved a bug that used to cause virtually limitless acceleration in the lower gears, when still engaged at high velocity and unlocked converter.
    - Within reasonable limits, the user can now, via throttle pedal “kickdown” and/or additional configuration (depending on current speed), force the gearbox to downshift prematurely, when switching from ‘D’ to one of the constrained modes. Limited severity and duration of potentially resulting slippage in that scenario. Overall, when the constrained modes are used, the gearbox now feels “more aggressive”.
136. Other transmission changes:
    - When the engine is off, yet a forward-driving transmission mode is in effect, throttling no longer causes first gear engagement.
    - When a transmission malfunction has occurred, neither automatic engagement and disengagement of the first gear at standstill, nor torque converter locking and unlocking occur any longer; both functions remain “stuck” in the state they were in at the time the malfunction occurred. The same applies to the internal state of the *TopoDyn* function. As a side effect of the former change, when decelerating while in the defunct state, and having an engaged gear and a locked-up torque converter, the engine will upon reaching standstill “get choked”, as if the transmission were manual.
137. Changes to dashboard transmission mode selectors, and interfacing between cockpit/VDV and transmission subsystems:
    - Cockpit is now solely responsible for button pushing logic and illumination. Like actual gear engagement, mode update request acknowledgment constraints, such as having to brake prior to requesting that the effective mode be changed, are now implemented and enforced on the transmission side.
    - Whether the ‘N’ selector remains engaged when no longer held down, or pops back up, is now configurable.
    - Whether, at standstill, braking is required in order to switch from ‘N’ to any other mode, is now configurable.
    - When moving forward, any forward-driving mode can now be selected (and be acknowledged unconditionally), even when currently in ‘N’.

- When quickly transitioning from any forward-driving mode to ‘N’ and then, prior to ‘N’ actually having been acknowledged, to ‘R’, as well as the other way around, the mode selected last will still eventually become effective, as long as the apposite constraints hold (e.g., not moving too fast for reverse gear engagement). Previously the transition would appear to be successful, yet actual gear engagement would not occur as expected (the system would “get stuck” in neutral).
- Implemented transmission mode selector blinking. Whether and which selectors are to blink, and at what intervals, in the event of a) malfunction, and b) the gearbox not being able to transition to, or retain, the mode requested by the user, are now configurable.
- The VDV display’s “gear engaged” warning and the dashboard’s yellow “master warn” indicator illumination now also account for ‘R’ mode selection.
- Consolidated “normal” and night-map textures used by the new selectors into two files (previously 3 copies of the same files used to be referenced without good reason).

138. Introduced a more flexible configuration scheme, to address some deficiencies in the prior approach of relying on .cti-variables alone.
    - Scripts now rely on reasonable hard-coded values by default. Constants can override these defaults for broad groups of vehicles using the same const-file. Further down the road, .cti-variables may selectively override the former on a per-repaint basis. Lastly, in-game / post-spawn, any of the above may be overridden “on-demand”.
    - This new approach also allows the default / unnamed repaint, for which normally no .cti-file is available, to be configured with sensible defaults.
    - All constants and variables participating in the reworked scheme can now be validated, boosting script predictability / robustness. When found invalid, the value at the next lowest configuration layer is considered instead, or, as a last resort, the bottom-most layer of “default-defaults”, which are always considered valid.
    - As an implicit result of the refactoring undertaken, several cases of flawed repetitive script-re-initialization (initialization logic erroneously being performed not only upon the vehicle’s initial spawning, but also upon subsequent re-spawning due to saved situation reload, leading, for example, to switches being reset back to their original positions post-reload) have been resolved.
139. Adjusted length of interior fluorescent light tubes to be roughly equal. The front-most pair of tubes was previously obviously too long, compared to the other two pairs in the middle and rear.
140. Converted all mod-specific textures to PNG. Some of the JPG textures previously in use would not be applied, according to beta-testers, presumably out of lack of support for JPG compression. Curiously the issue could never be reproduced locally.
141. Removed pseudo-shadows from below IBIS / ticket printer and cash desk, originating from the vanilla Citaro.
142. After opening and subsequently resetting the door emergency opening valves, the usual preconditions (electrics and pressure availability, as well as, in the case of opening, maximum allowed velocity) once again apply. Previously, after door re-pressurization, it was possible to trigger adjustment even if those preconditions were not met (e.g. when electrics off).
143. Changes related to stop brake release:
    - The stop brake will no longer release when ECAS leveling is in progress.
    - The “NOT-AUS” switch is now toggle-able. It is only acknowledged when toggled and the throttle pedal is additionally pushed. When acknowledged, any stop brake release preconditions are disregarded, and the stop brake is instantly released. The overridden constraints include open doors (front door depending on configuration), the wheelchair ramp (when unfolded), kneeling (when in progress or completely knelt down), and ECAS

leveling (when in progress); they *do not* include the manual ("20-h") stop brake engagement switch. The vehicle will still kneel back up automatically, if previously knelt down.
- When electrics are off or unavailable, unless the manual ("20-h") switch is on, the stop brake will be released without throttling.
- The related VDV warning (item 87) is no longer triggered by the "NOT-AUS" switch's toggling per se, but by any of the aforementioned stop brake release constraints actually being violated *as a consequence* of the switch's toggling; if the switch, for example, causes the stop brake to be released while the rear door is open, the warning will be displayed until either the rear door gets closed, or the stop brake gets re-engaged.

144. Dynamically-assigned vehicle identifiers (license plate registration number and operator-specific number) are now visible again by default (configurable, disabled for CSB-native repaints). As a consequence, those identifiers are now correctly applied to "foreign" repaints, which consequently no longer appear weird because of having a blank license plate. This is – at least until further CSB variants make it into the mod – a necessary provision in order to properly support registration of multiple instances of the modded vehicles with AI-lists, i.e. of multiple repaints, given that the CSB-native ones have static identifiers.

Non-functional changes

145. Eliminated unused legacy / pre-VDV standalone electrics key and engine starter/stopper button logic, as part of the ongoing effort towards cockpit simplification / consolidation. Removed `cp_key_rot_drag`, `cp_schluessel_mov_drag`, `kw_batterietrennschalter`, and `cp_batterietrennschalter_toggle` cockpit triggers; the `schluessel_frame` cockpit macro, the `automatic_battery_switch` cockpit constant, and all related variables not otherwise relied on any longer. The sole remaining relevant triggers are those introduced by Morphi's rework to the engine script, i.e., `engine.osc:cp_batterietrennschalter_toggle` (increases electrics mode, 'E') and `engine.osc:kw_m_enginestart` (decreases electrics mode, 'M'). These triggers have been relocated from the engine back to the cockpit script, retaining their "historical" names for user convenience.
146. Revised structure and inline documentation of ZF const-file.
147. Documentation and distribution archive changes:
    - Changed root texture path from `Texture\up` to `Texture\ACE`, for consistency's sake.
    - Added `Model\ACE\O530\imported` and `Sound-Citaro\ACE\imported` sub-directories for distinguishing modification-own or redistributed content from non-redistributed dependent content.
    - In documentation and implementation alike, variable names included, replaced all occurrences of "front fog lights" with "head fog lights", once again for consistency's sake.
    - Readme:
        - Added "AI dirtiness" section.
        - Added *Florito* to contributors.
        - Split off "trigger reference" from "configuration" (formerly ".cti-variable reference") section; organized the latter by subsystem, and added introduction on revised configuration scheme.
        - Made introduction to "AI lighting diagnostic mode" section a tad more thorough.
        - Simplified format of "Known issues" and "Changelog" sections, eliminating, along the way, some instances of excessive detail that would only bore the reader to tears.
        - Added disclaimer entry to "Known issues" section, about the function of saved situation reload currently being unsupported.

- Added erroneously absent `bus_doorfront0_ext` trigger reference (item 119).
- Added uninstallation instructions.