



miniSim™

Driving Simulators for Research



2020

Document Version 28

National Advanced Driving
Simulator University of Iowa

NADS-miniSim@uiowa.edu

© Copyright 2020 by National Advanced Driving
Simulator, The University of Iowa. All rights

NADS History and Overview

The National Advanced Driving Simulator ([NADS](#)) is a self-sustained transportation safety research center in the University of Iowa's College of Engineering. Funded by government and industry, NADS utilizes its suite of world-class driving simulators and instrumented on-road vehicles to conduct research studies for the private and public sectors. NADS is staffed by more than 40 faculty, staff, and students who specialize in studying the connection between drivers and vehicles.

NADS Organization

The NADS is an independent, self-funded research unit at The University of Iowa. External contracts fund employment of all staff, graduate and undergraduate students. The facility is also operated, maintained and upgraded in a self-sustained manner. The [miniSim™](#) is a program within NADS that leverages technologies developed at NADS to provide solutions for research and other applications.

The NADS organization is part of the [Iowa Technology Institute \(ITI\)](#), a multidisciplinary research center within the [College of Engineering](#) that specializes in conducting applied research in modeling and simulation. NADS actively collaborates with other UI entities including the Public Policy Center and the Colleges of Medicine, Pharmacy, Liberal Arts, and Public Health.

NADS shares with ITI and The University of Iowa a commitment to academic advancement, technological innovation, and the transfer of research results to industrial and governmental communities. NADS is also committed to supporting the activities of local economic development initiatives as well as local STEM (Science, Technology, Engineering, Math) initiatives.



Figure 1 – The NADS-1 Driving Simulator





Figure 2 - NADS and miniSim current and past partnerships

miniSim™ Introduction

The miniSim™ driving simulator is ideally suited to applied R&D, academic, and clinical research. Backed by decades of development at the National Advanced Driving Simulator (NADS) at the University of Iowa, the miniSim is both sophisticated and affordable.

[Overview](#)

[Examples](#)

The powerful scenario control and data acquisition capabilities give you the flexibility you require, and all the data from every run at your fingertips. The miniSim is a result of over 20 years of research and development and is in use daily by our researchers and staff.

The hardware is available in three configurations: Half-Cab, Quarter-Cab, Simplified Cab, and Desktop. The software is the same for all versions and also supports vehicle automation applications, custom cab designs, and user-developed subsystems. NADS technology is used by NADS staff daily to fulfill scientific research contracts on our NADS-1, NADS-2, and miniSim simulators, ensuring that you get the most up-to-date capabilities. A common code base and tools are used across all our simulators, so a scenario developed on the miniSim will run on other miniSim systems, or NADS-1/NADS-2 and vice-versa.



Figure 1: The miniSim™ Driving Simulator with Partial Cab, and Operator Display.

The miniSim consists of four core tools:

1. **Tile Mosaic Tool (TMT)** for assembling a road network from a library of over 250 road/landscape segments called 'tiles'. You use the TMT to connect them and export a complete database to the ISAT and miniSim. NADS can also create new tiles or modify existing ones to meet your specific needs.
2. **Interactive Scenario Authoring Tool (ISAT)** for building scenarios on the road network. ISAT is GUI-driven and does not require scripting. Has edit, rehearse, and playback modes.
3. **miniSim Core** runs the scenario on the assembled database and provides both real-time measures and a comprehensive data acquisition file in a secure binary format for post-processing.
4. **nDaqTools** are interfaces for MATLAB used to control the data reduction process, including the definition of the measures used. Text output is also available for Python users.

Software functions include:

- **ACC and Lane Following** systems are commonly used to emulate level 2 automation and hand-off of control. ACC parameters are adjustable including full-range speed control.
- **LKA and LDW** systems are built-in, and parameters are adjustable. The LKA system provides torque cues through the steering loader.
- **NADS AutoDriver** is an automated driving capability incorporated into the NADSDyna vehicle dynamics software to take over operation of the steer, gas and brake controls to autonomously follow a route through the world. The characteristics of the 12 available behaviors (lane change, turns, etc.) are controlled through a set of parameter files and a command vocabulary that may be given through the following mechanisms:
 - Scenario action (triggered by scenario event)
 - Manual action (button press)
 - External input via UDP
- **Virtual World API** is a programmer interface to extract road and scenario-object information in real-time when provided with the location of the driver's own-vehicle.
- **External Control** gives the advanced user remote control over the driver controls for autonomous system testing and development. Currently UDP interface. ROS interface under development.
- **Web Socket and UDP** interfaces allow integration with external systems and displays, such as Haptic Seats, touchscreens, etc.
- **Analog and Digital I/O** is used to integrate driver controls and also trigger or synchronize external devices such as alerts or physiological data collection systems (D-Lab, Biopac, EEG, etc).
- **User-Defined Subsystems** are user-written programs that run within the miniSim real-time environment and have read/write access to the shared memory space.

Available accessories:

- **Motion Base** provides roll, pitch, and heave onset cues for improved immersion and vehicle control. Can easily be added to any miniSim without an increase in height or footprint.
- **Infotainment** systems emulate OEM-style infotainment interfaces while providing data acquisition and operator control.
- **VidCap** video recording system captures 4 channels of HD digital video synchronized with the simulation
- **Springfield Road Network** has 230 miles of roads covering 285 sq. miles! Complete and ready to use with initial settings for traffic, signal timing, and signs
- **Haptic Seat** 6 programmable transducers in the seat pan. Higher-channel counts available

We provide the following services:

- Prompt, in-depth user support
- Custom scenario development
- Custom environments and models
- Turn-key experiments
- Hardware Upgrades
- Simulator design and development
- Refresher Training
- Consulting



Figure 2: High-Resolution Rendering of Typical Scene

miniSim™ Community

A growing community of researchers are using NADS simulator technology in their programs as shown in the map below.



Why miniSim™?

- ✓ The best value among research simulators
- ✓ Unmatched support by our own developers and users
- ✓ Custom scenarios, data reduction, road networks, and hardware solutions are available to meet your needs exactly
- ✓ Access to a national network of researchers who are using the same core simulation technology, facilitating standardization, collaboration and sharing of scenarios, data, and ideas.
- ✓ NADS conducts ground-breaking driving simulation research, allowing us to maintain state of the art simulators with continued updates for current research topics
- ✓ NADS simulation platforms represent nearly 25 years of simulation engineering, longer than any other driving research organization

Support and Maintenance

The NADS Software are delivered and supported by NADS in exchange for a yearly Software Support and Maintenance Fee (“Annual Fee”). The Annual Fee is required for use of the software and for support, updates, bug-fixes, and new releases.

NADS will invoice for the support and maintenance fee upon contract execution, and annually thereafter on the date on which the system was delivered and initial user training completed (‘renewal date’). Upon receipt of this payment, NADS will update the user’s USB keys (Keylok) for another year.

NADS retains the right to adjust the Annual Fee and agrees to notify Recipient one (1) month prior to the automatic renewal date of fee adjustment. See miniSim Terms and Conditions for more details.

Support is available from 9am-5pm Central Time Mon-Fri, except for University Holidays†. Support is via email, phone, or remotely via the miniSim [MyQuickSupport](#) application. NADS will acknowledge support requests within 24 hours. All users have access to training videos and materials on the miniSim website.

Hardware (cab, PCs, displays, etc.) is supplied as-is and without warranty, except for manufacturer’s warranty if applicable. Support and repairs will be mutually agreed upon and provided on a non-profit basis.

Custom software and hardware development is available, please contact the miniSim team for a quote.

†University Holidays are: New Year’s Day, Martin Luther King Day, Memorial Day, Independence Day, Labor Day, Thanksgiving Day, the Friday following Thanksgiving, Christmas Day, and an additional day near Christmas designated in the official University calendar.

How Do I Get Support?

The primary method is via email: miniSim-Support@uiowa.edu

We check this account at least once a day, and delegate requests to the appropriate team member. We acknowledge the requests within 24 hours (not including weekends and Holidays), and will resolve the issue quickly in the order they come in.

If you send email from your personal account, please state where you are located. We use this information to determine what system you have (PC specs, Software versions, etc) and keep track of issues.

Online Support Materials

Visit the [miniSim Support Page](#) for the most up to date support information, training videos, and manuals.

Also, on your mininSim, there is a ‘Manuals’ shortcut on the desktop, or in the miniSim folder (typically **C:/NadsMiniSim_2.x.x/manuals**)

What is not support?

Support is not Training. During training the user is oriented to the system architecture, how the modules work together, system operation, tips on good practices, practice using system, etc. We provide training to all new users, and we expect that they pass-on this information to new students/researchers as they enter the lab. We provide 'refresher' training remotely (web meeting) for a reasonable fee, and there is no restriction on the number of students. A few days of intensive training will make your team much more productive and confident in using the miniSim. Ask us for a quote!

Support is not Development. We are pleased to hear requests for new features; the NADS team will evaluate feature requests for inclusion into a future software release. If specific new functionality is desired in a specific timeframe, please contact the miniSim team for a quote.

Backups

Users are responsible for maintaining the miniSim PC and archiving collected data. This includes Windows Updates, backups, data archive/storage, security, and PC repairs. Your on-site IT support can typically handle these areas easily. Most Universities offer cloud storage services at reasonable cost for long-term storage of research data (some sponsors may require this).

Note that the miniSim PC is usually supplied with two hard drives in a mirrored array (RAID 1). This provides redundancy in the case of a failure, but this is not intended for long-term secure storage.

Hardware

Hardware (cab, PCs, displays, etc.) is supplied as-is and without warranty except for manufacturer's warranty if applicable. Support and repairs will be defined by NADS and provided on a non-profit basis.

We will aid in troubleshooting hardware problems and locating replacement parts. In most cases we can tell you what part to buy directly, as it will be more expensive for us to source them for you. If necessary, we will provide a quote for miniSim-Specific parts, or for a site visit to troubleshoot and repair the system. For hardware upgrades, we can provide you with a quote for a complete PC from NADS.

Network Access

The miniSim software does not require internet access to run, but internet (WAN) access is highly desirable for several reasons:

- **Windows updates.** The miniSim PC runs Windows 10 64-bit Pro. Windows will check for and download updates automatically. Windows Defender antivirus and Windows Firewall are used. We recommend that users update their system and virus definitions regularly but be aware that updates to drivers (eg video drivers) may affect system operation.

We do recommend disconnecting the system from the external internet (WAN) during data collection. This will eliminate the possibility that updates will be downloaded and applied while the miniSim is running, which may affect the smoothness of the graphics rendering.

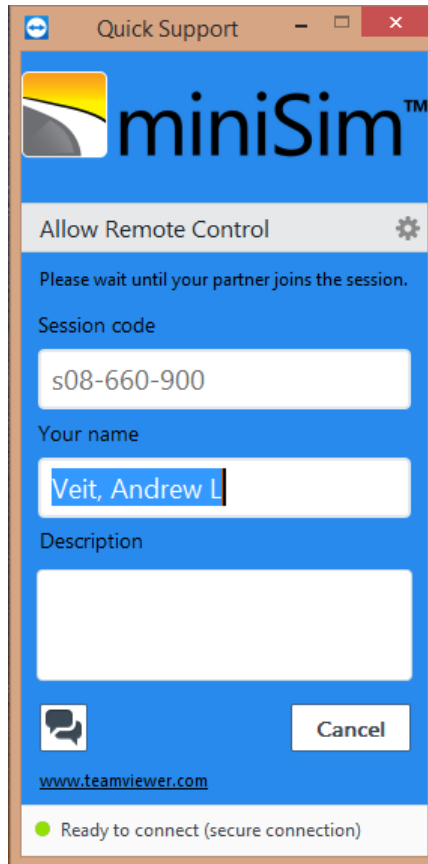
- **User support and software updates.** Internet access is **highly** desirable. Remote Support is described in detail below. Many users will connect their miniSim to the internet only when they need remote support.
- **Data backup.** The use of network storage is recommended. The miniSim uses a dual-disk RAID (mirrored) for redundancy, but backup and archiving of experimental data is the user's responsibility. Most Universities provide cloud data storage at reasonable cost, and many users have local network attached storage (NAS) on their laboratory local area network (LAN).
- **Data analysis.** Most researchers will conduct data analysis on their personal workstation or laptop to keep the miniSim available for scenario development and data collection. Use of shared network drives, a NAS, or cloud storage makes the process of moving and archiving data files much easier.

Remote Access via *QuickSupport*

NADS has chosen TeamViewer [QuickSupport](#) as a remote access tool for miniSim user support. This has the following benefits:

QuickSupport provides these benefits:

1. QuickSupport is a miniSim-branded executable that the user runs *only* when they would like us to log into the system. It is a single executable (TeamViewerQS.exe) that does not auto-launch at boot like the 'free' or 'Host' TeamViewer versions, thus we cannot log into unattended systems.
2. Only members of the NADS miniSim support team can log into the system when granted permission by the user at each occurrence (white list). We are not able to see or log into the system without the QuickSupport application running and the user present to grant access.
 - a. Completely unattended support is **not** possible with QuickSupport. If you require unattended access by our support team, the TeamViewer 'Host' is used. This is also a white-list application, only allowing miniSim team members access.
3. Makes upgrades smoother. Beyond simple software patches consisting of a few files, applying updates and migrating a user's experiments from an old miniSim install to a new one may require assistance. An 8-chapter video that describes the migration process is available [here](#).
4. Many users add additional devices to the miniSim, such as eye-tracking or touch-screen tasks. In these cases, remote access greatly simplifies configuring communication with the miniSim.
5. Unlike Windows Remote Desktop ('RDP'):
 - a. Support personnel can view all the miniSim desktop displays (5 or more). RDP only handles a single display.
 - b. The user can observe what we are doing. During RDP the user's screen is black.
 - c. Has integral chat and file-transfer functions. No chat in RDP.
 - d. When the QuickSupport session is terminated, the desktop layout is un-affected (RDP requires a re-boot to return the multi-display desktop to its former configuration).



miniSim TeamViewer QuickSupport

Shipping, Installation and Training

Two transportation options are available, and in both cases the customer pays the carrier directly:

Commercial Padded Mover is preferable and cheaper when the destination is in the lower 48 states. NADS boxes the PC and peripherals with plenty of cushioning and protects the cab with stretch wrap. The moving company will unload and deliver the system to the desired room at its destination, ready for installation.

Less Than Truckload (LTL) Freight requires the simulator cab and accessories are packed in a sturdy wood crate(s) to protect them from damage. There is an extra expense for crating, and a liftgate for pickup. Check with your facilities team about freight deliveries, loading docks, and storage if you are considering LTL freight.



Figure 1 – Shipment by LTL Freight (L) or Commercial Padded Mover (R)

Installation and On-Site Training

Installation is performed by a NADS engineer at the customer's site for Quarter and Simplified Cabs. This includes unpacking, connecting cabling, and monitor alignment. This can take up to 1 day.

Training is provided to thoroughly cover simulator start up, operation, shut down, use of the miniSim software and other software tools. This training lasts 1 to 1 ½ days and covers:

- NADS miniSim Software and Hardware
 - MiniSim hardware and connections
 - Software Architecture and Workflow
 - File locations and file descriptions
 - Importing Scenarios and Running the MiniSim,
 - Data Collection Mode and Experimental Configuration File
 - MiniSim Variables and Definitions
 - Report Mode, Runtime Measures, Events, and Log Streams
 - Configuration files (viewports, digital and analog inputs).
 - Calibration of inputs, validation
- NADS Tile Mosaic Tool (TMT)
 - Assembly of road network (world) using tiles

- Viewing a tile without using miniSim
- Use of the Level of Detail (LOD) function
- Publishing and installing the completed world to MiniSim
- Intro to ISAT Functionality and Methodology
 - Simulation Framework
 - Edit, Rehearse, Playback Mode
- Basic Simulator operation
 - Start up and shutdown, scenario selection, start and stop simulation
 - Import scenarios and running them on the miniSim™
 - Viewing the data file collected on the miniSim in the ISAT tool
 - Using **daqConvert** utility to convert the data files into MATLAB or Text format

Remote ISAT Training

The remainder of the training will be remote via a Web Meeting. There is no limit to the number attendees, and it will take 3 sessions and will primarily cover the Integrated Scenario Authoring Tool (ISAT):

- Overview of ISAT Functionality and Methodology
 - Simulation Framework
 - Edit, Rehearse, Playback Mode
 - Scenario Coordinators (triggers and actions)
 - Static Objects
 - Dynamic Objects
 - Autonomous Objects
 - Grouping and External References
 - Scenario Authoring Methodology
 - Tips for Robust Scenarios
 - Log Streams
- Step by Step instructions to build the following scenario events:
 - Pedestrian Incursion Scenario
 - Right Incursion Scenario
 - Right Incursion Scenario
 - Yellow Light Dilemma
 - Interstate Driving event
- Audio and Visual Features
 - Playing Sounds
 - Displaying Text
 - Virtual Objects

NADS nDAQTools – Data Reduction with MATLAB is covered in a [10-chapter video](#)

- Configuring MATLAB for use with nDAQTools
- The Disposition Spreadsheet
- Developing Reduction Scripts
- Reduction Examples

Additional materials are available on the miniSim [Video Training and Knowledge Library](#).

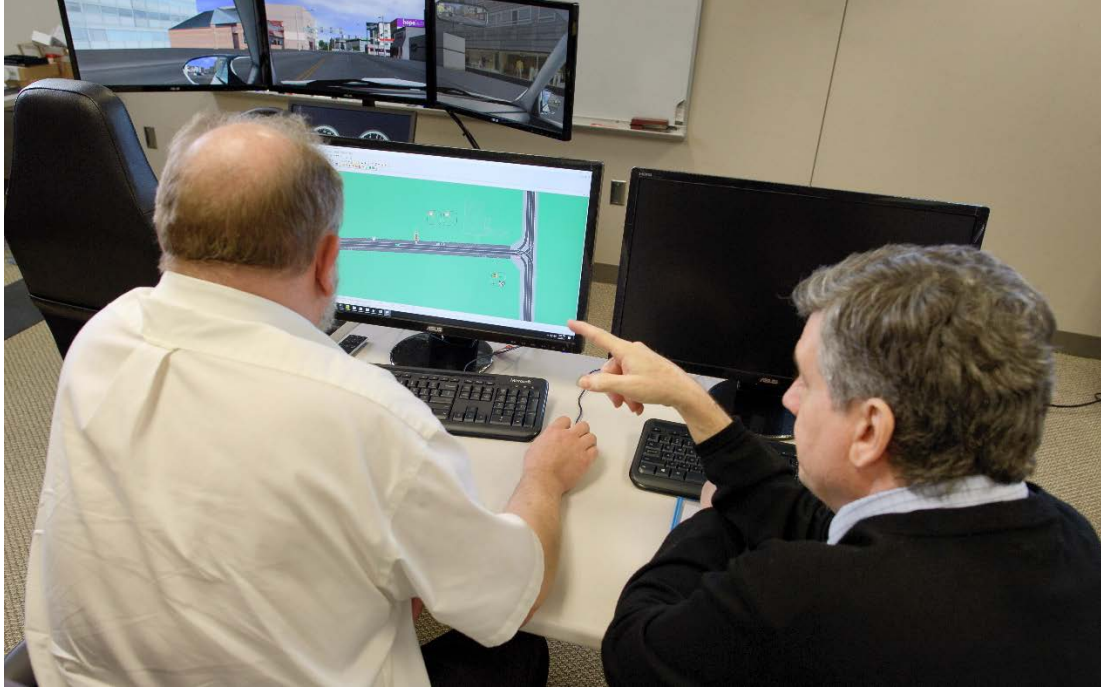


Figure 2: ISAT Training Session

Other Resources

A user-publication list is available upon request.

NEW! Simulation “Boot Camp”. All 5 webinars on the SAFER-SIM channel. This link will take you to the first video in the playlist with access to the others:

https://www.youtube.com/watch?v=uMb3CqVZD_w&list=UUE8CN3JX8_mkAf8d8-UPzKQ

The *Handbook of Driving Simulation for Engineering, Medicine, and Psychology* was written by researchers in the community about the technology, best practices, etc.:

http://www.nads-sc.uiowa.edu/workshop/wkshp_resources.php

NADS Annual Reports:

[2019 Annual Report](#)

[2018 Annual Report](#)

Materials for an introduction to driving simulation session TRB in 2013:

<http://www.nads-sc.uiowa.edu/workshop/>

NADS publications:

<http://www.nads-sc.uiowa.edu/publications.php>

<http://www.nads-sc.uiowa.edu/brochures.php>

NHTSA Publications:

<https://www.nhtsa.gov/research-data>

Behavioral: <https://www.nhtsa.gov/behavioral-research>

Human Factors: <https://www.nhtsa.gov/research-data/human-factors>

Standardization of measures used in the driving simulation community is an ongoing process. A new SAE Recommended Practice is available to address this issue:

[SAE J2944 Operational Definitions of Driving Performance Measures and Statistics](#)

A draft version of the Practice is available here:

http://www.auto-ui.org/docs/sae_J2944_PG_13-02-12.pdf

http://www.auto-ui.org/docs/sae_J2944_appendices_PG_130212.pdf

These references will be helpful to those developing data reduction scripts.

Tier 1 University Transportation Center (UTC), SAFER-SIM

<http://safersim.nads-sc.uiowa.edu/>

Videos of the NADS-1 and miniSim available on Youtube

<https://www.youtube.com/user/NADSITGroup>

User-Developed Python Data Reduction Tools (‘undaqTools’ from the University of Idaho):

<https://github.com/rogerlew/undaqTools>

User-developed tools from The Sonification Lab at Georgia Tech University:

Lab Page: <http://sonify.psych.gatech.edu/research/driving/index.html>

HUD Designer: <https://smartech.gatech.edu/handle/1853/54041>

STING Telemetry Data Module: <https://smartech.gatech.edu/handle/1853/54040>

Sim Sickness Screening Protocol: <https://smartech.gatech.edu/handle/1853/53375>

Training Manual: <https://smartech.gatech.edu/handle/1853/53374>

User Videos

University of Toronto

<https://www.youtube.com/watch?v=7tlYIUPTnXg>

<https://vimeo.com/164645263>

https://www.youtube.com/watch?v=Sf0h8_V9toM

University of Idaho

<https://www.youtube.com/watch?v=glQIFc927XY>

Yale University

<https://www.youtube.com/watch?v=XTwTn9h1qyA>

University of Kansas:

<https://www.youtube.com/watch?v=vAhCr6XosVI>

Harman International:

<https://www.youtube.com/watch?v=AxHdLRxm7eg>

Idaho National Lab:

<https://www.youtube.com/watch?v=8uIO10Hk09E>

User Websites and Media:

University of Kansas:

<http://www.newswise.com/articles/new-driving-simulator-lab-accelerates-research-into-driver-behavior-and-vehicle-technology>

University of Central Florida:

http://www.baynews9.com/content/news/baynews9/news/article.html/content/news/articles/cfn/2017/1/30/central_florida_part.html

University Of Windsor:

<http://www.uwindsor.ca/dailynews/2015-05-31/funds-provide-income-transportation-science-and-engineering-students>

Georgia Technological Univ.:

<http://www.gpb.org/news/2014/11/19/driverless-cars-state-legislators-make-recommendations>

Univ. of Central Florida:

<http://spacecoastdaily.com/2015/03/ucf-studying-ways-to-keep-drivers-safe-in-thick-fog/>

Driving MBA:

<http://www.newsweek.com/2015/04/10/ensuring-your-childs-drivers-license-isnt-license-kill-317487.html#.VSCBmA2FvJg.email>

University of Iowa:

<http://www.engineering.uiowa.edu/news/ui-driving-simulator-help-test-new-artificial-lens-cataract-patients>

~END~

miniSim™ Quarter-Cab Driving Simulator



Figure 1: The miniSim™ Driving Simulator with Partial Cab, and Operator Display.

The Quarter-Cab simulator is provided with the following:

- miniSim Quarter-Cab with the following features:
 - LCD display virtual instrument cluster
 - Instrumented steering wheel controls for Adaptive Cruise Control, Lane Keeping Assist, and Lane Following. 4 buttons for user-defined driver response.
 - Instrumented Turn Signals, Horn, Response Buttons, Gear Selector, Headlights, Wipers, Mirror Controls, and user-defined buttons
 - Real-vehicle seat, steering wheel, and pedals
 - Active DirectX steering loader with optional 20 N*m torque sensor
 - 2.1 channel sound system with vibration transducer
- NADS miniSim computer loaded and tested prior to shipment
- Three 1080p LED LCD widescreen displays of 48" diagonal measurement
 - Horizontal FOV: 138 to 180 degrees
- A 24" operator LCD for miniSim, ISAT and TMT interfaces
- Software as described in the NADS miniSim™ Software and PC section

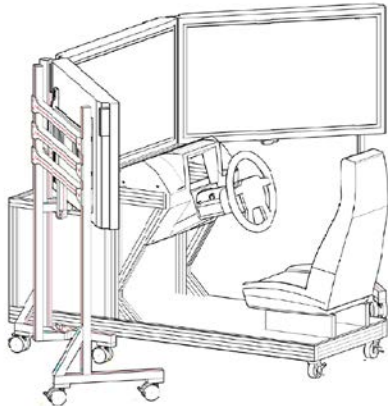


The miniSim™ PC Rack. Shown with Optional Video Capture PC



Available miniSim™ Driver Interfaces

miniSim ¼ Cab Dimensions



Cab:

Overall Dimensions:

Length: 67" / 170 cm min, 75" / 191 cm max

Width: 27" / 68.6 cm

Height: 41" / 104 cm

Electrical:

120 VAC, 60 Hz

750 W (cab and PC)

Monitor Stand

Field of View (at 48" / 122 cm viewing distance):

141° Horizontal

27.5° Vertical

Display Type:

48" / 122 cm LED Active Backlit LCD (1920x1080)

Display Dimensions:

43.0 x 25.4 x 2.6" / 109.2 x 64.5 x 6.6 cm

Overall Dimensions:

Width: 120" / 30.5 cm (3-display)

43" / 109.2 cm (1-display)

Height: 58" / 147 cm

Weight: 220 lb. / 100 kg (3-display)

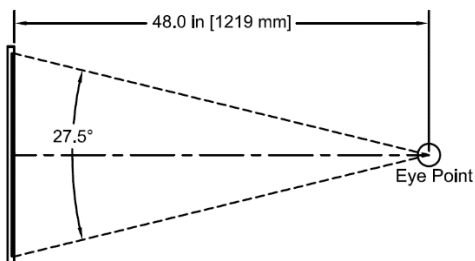
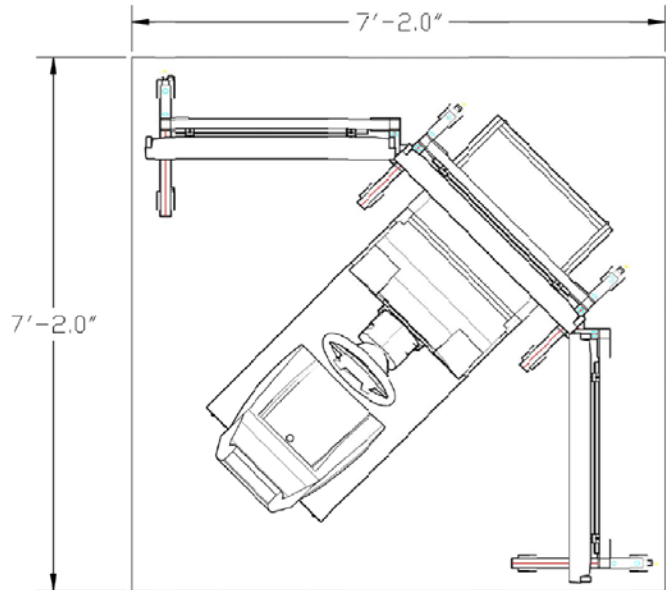
75 lb. / 34 kg (1-display)

Electrical:

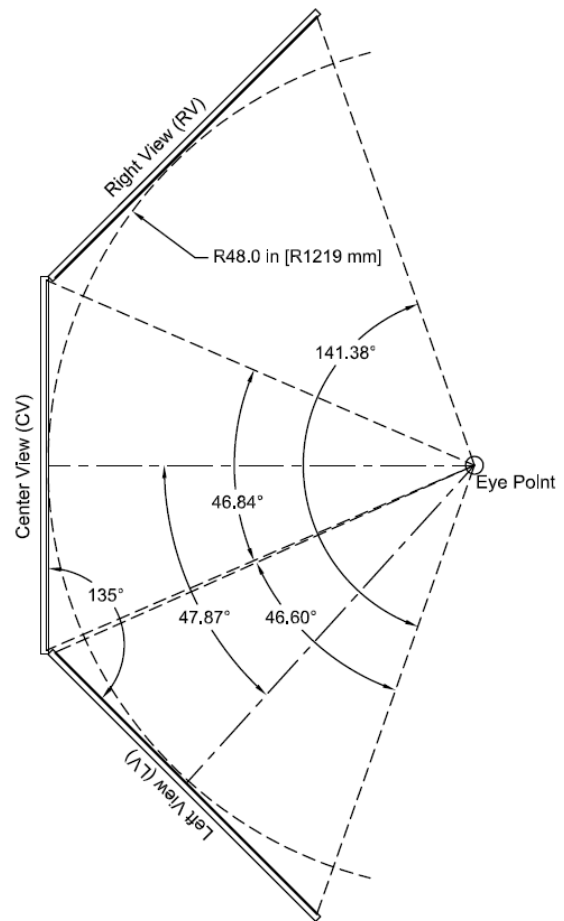
19.5 VDC; Power Adapter: 100 - 240 VAC, 50-60 HZ

225 W / 1.5 W standby (3-display)

75 W / 0.5 W standby (1-display)



Display Geometry (side view)
~27 Degree Vertical FOV



Display Geometry (top view)
Quarter-Cab/Simplified-Cab MiniSim
48" LED LCD Displays, at 45.0 Degrees
~140 Degree Horizontal FOV

miniSim ¼ Cab Hardware Description

Displays

The ¼-cab simulator uses three LED backlit LCD displays. Other display technology options are available including the using of smaller screens, and projection systems.

- Screen size 48" (each display)
- Resolution 1920x1080
- Refresh rate 60Hz
- Horizontal FOV 141 degrees (up to 180 degrees)¹
- Vertical FOV 27.5 degrees²

Operator Controls

The simulator features an actual vehicle steering wheel and pedals controls. The steering wheel uses a semi-active controller that is driven by a DC motor and microcontroller. The steering column features a functional turn signal indicator. The brake and accelerator pedals feature a realistic feel. Force feedback on the brake is available as an option. Other switches can be instrumented per customer requirements.

Hardware controls:

- Steering Wheel and Buttons (Cruise, ACC, LKA, and response)
- Accelerator pedal
- Brake pedal
- Gear Selector
- Turn signal
- Seat
- Wipers
- Hi/Lo Beams
- Hazards
- Horn

Functioning gauges and lights on virtual instrument cluster:

- Speedometer
- Tachometer
- Gear selector
- Turn signals

Computers

The simulator uses a single computer to control all aspects of the real-time simulator system. The PC uses off-the-shelf parts for easy maintenance and upgrades.

¹ Variable with how close the displays are placed to the driver's eye-point

² Variable with how close the displays are placed to the driver's eye-point

miniSim™ Simplified-Cab Simulator



Figure 1: Simplified Cab shown with three 24-inch LCDs

- NADS miniSim computer loaded and tested prior to shipment
- LCD dashboard display for the instrument cluster
- Automotive seat
- Tilt wheel adjustment
- Fanatec Club Sport V2.5 base, Full-Size Leather-Wrapped Wheel, CSR Elite Pedals
- Fanatec CSL Elite All-Metal Pedals with Load Cell option
- Fanatec Load Cell Pedal Option
- 12 Wheel buttons that can be used for driver response, look left/right, shoulder check left/right, ACC, Lane Keeping Assist, and AutoDriver
- Labeled buttons for Engine Start, Gear Select, Wipers, Hazard, Mirrors, Headlights, and two Auxiliary Buttons
- Tactile transducer and amplifier provides road/engine vibration
- A 24" operator display with USB Extension for keyboard and mouse
- Software as described in the NADS miniSim™ Software and PC section

The following options are available:

- Three front 24 inch monitors
- Three front 48 inch monitors
- Single display (48 inch to 65 inch or more)
- Standing Desk

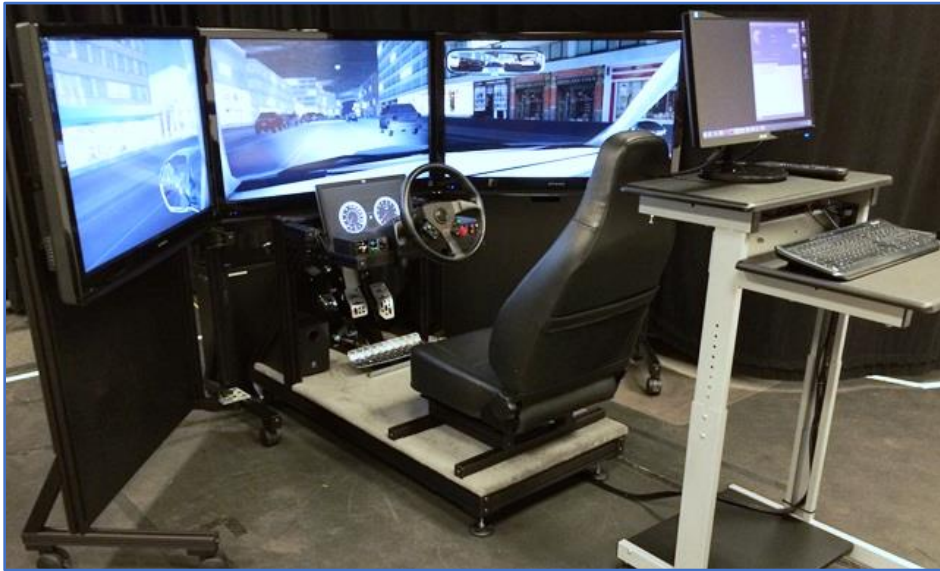


Figure 2: Simplified Cab shown with triple floor standing monitor stand, 46"-inch LCDs, standing desk



Figure 3: Simplified Cab shown with single monitor stand, 46" LCD, standing desk



Figure 4: 13" Leather-Wrapped Wheel and Button Boxes with Labeled Buttons



Figure 5: Fanatec ClubSport Wheel Base, CSL Elite Pedals, and 13" Steering Wheel with 6 Buttons
(note: paddles are removed)



Figure 6: Wheel buttons and Button Assignments

miniSim™ Desktop Driving Simulator



Figure 1: Desktop miniSim™ Driving Simulator with single 42" Display (L), and optional three 20" LCD Displays (R).

A Desktop miniSim driving simulator is shown in Figure 13. The simulator is provided with the following:

- NADS miniSim computer loaded and tested prior to shipment
- Three 24" front channel displays, and 16" dashboard display.
- Fanatec Club Sport V2.5 base, Full-Size Leather-Wrapped Wheel, CSR Elite Pedals with Loadcell Upgrade
- A 24" operator display with USB Extension for keyboard and mouse
- 2.1 Audio System
- Tactile Transducer (provides road/engine vibration)
- Software as described in the NADS miniSim™ Software and PC section

Note: Desk *not* included

The following options are available:

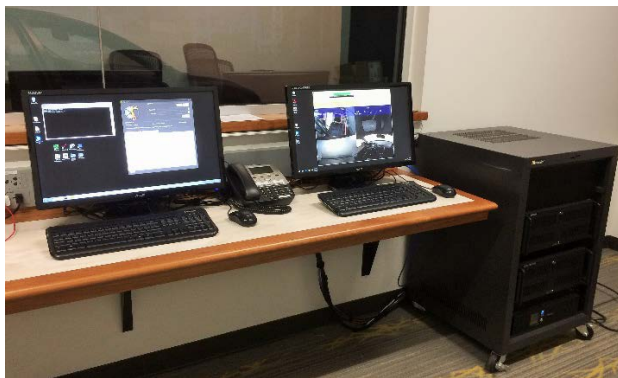
- Three front 24 inch monitors
- Three front 48 inch monitors
- Single display (48 inch to 65 inch or more)
- Standing Desk
- Button Boxes with Labeled buttons for Engine Start, Gear Select, Wipers, Hazard, Mirrors, Headlights, two Auxiliary Buttons
- ECCI Trackstar 6000 wheel (350mm wheel, 250-deg rotation only, passive spring-damper)

miniSim™ Half-Cab Driving Simulators

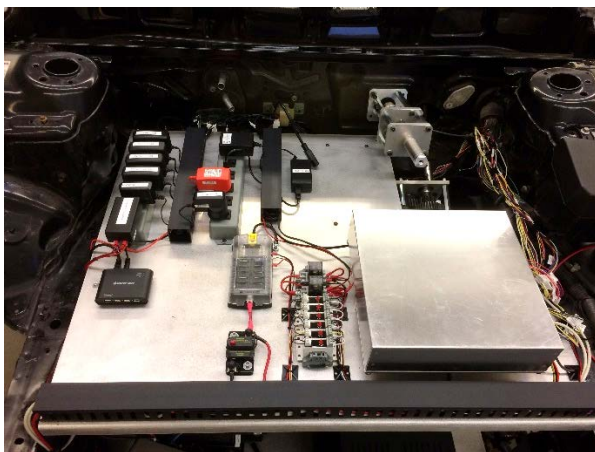


The Half-Cab simulator is equipped with the following:

- Fabricated from a Mazda 6 or Toyota Camry
- Driver Controls:
 - Active DirectX steering loader with 20 N*m torque sensor
 - Passive brake pedal loader of NADS design with load cell
 - OEM accelerator pedal displacement
 - LCD display virtual instrument cluster
 - Instrumented steering wheel controls for Adaptive Cruise Control, Lane Keeping Assist, and Lane Following. Two wheel buttons for user-defined driver response.
 - Instrumented Turn Signals, Horn, Response Buttons, Gear Selector, Headlights, Wipers, Mirror Controls.
 - Power leather seats, leather-wrapped steering wheel, tilt wheel
- Audio: 4 channel, 1 subwoofer, and 2 vibration transducers.
- Rearview mirrors are intact. Rear screen provides images for mirrors.
- OEM audio system works; sound mixed with simulator audio.
- Stock fan speed control
- NADS miniSim computer loaded and tested prior to shipment
- Computer Rack with UPS (space for optional NADS VidCap system)



Operator Control Location Examples



***Cab Instrumentation under Hood (left), and
Cab Connections and Controls in Passenger Wheel-Well (right)***

Projection Screen Types

Two general types of projection screens are available:

- Triple flat screens with images butted together
- Cylindrical screen with image warping and blending

The horizontal field of view is determined by many things, including screen size and viewing distance. Three flat screens can provide up nearly 180-deg Horizontal Field of View (HFOV). With a cylindrical screen and widescreen projectors, well over 180-deg HFOV can be delivered.



Three Flat Screens Installation (170-Deg HFOV)



Curved Screen Installation (166-deg FOV shown)



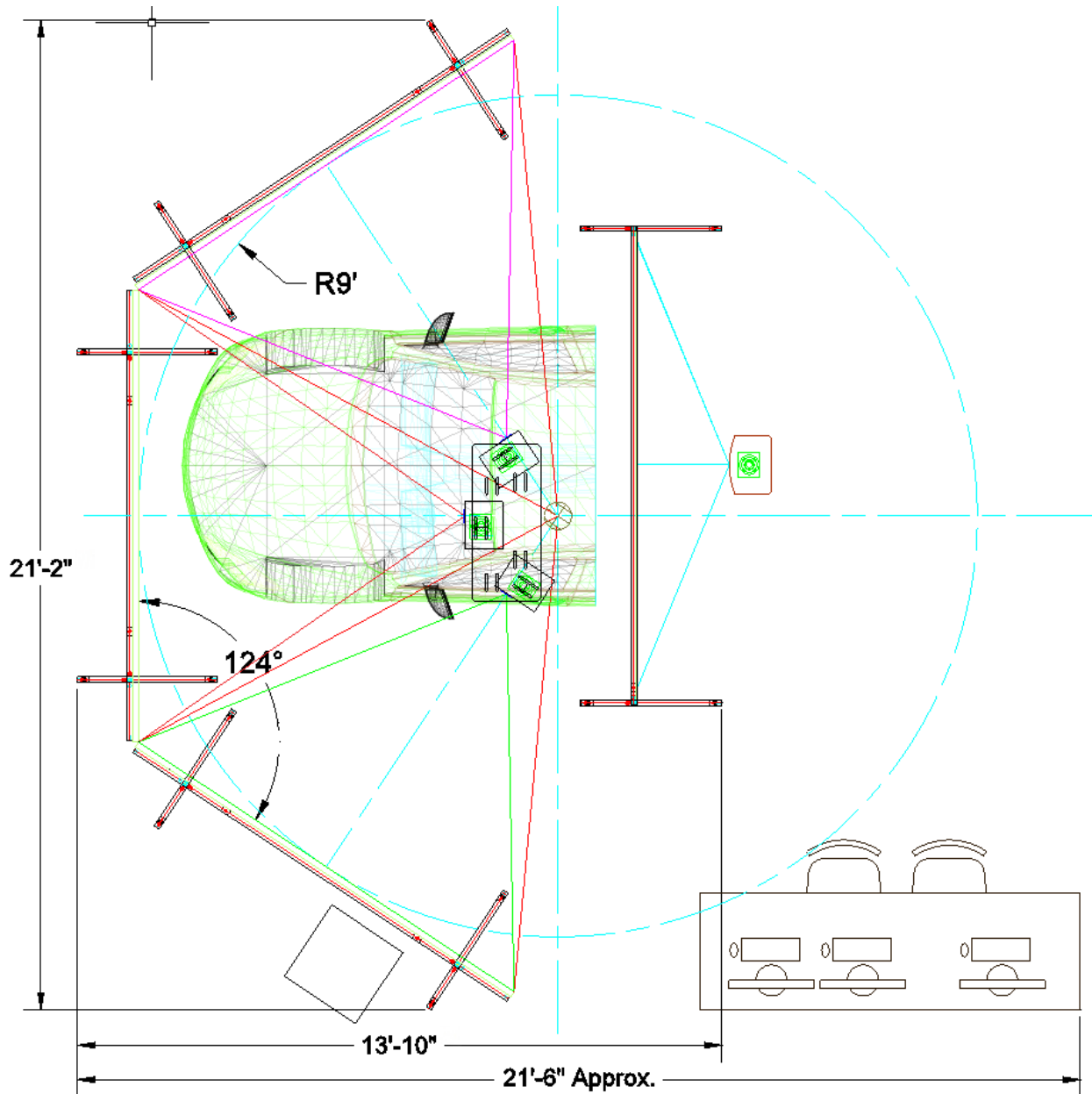
Photo Showing Rear Screen. Note rear image has 3 viewports: one for each mirror



Rearview mirror views, using stock vehicle mirrors

Triple Screen

The flat screens used in this system do not have borders, allowing the image to extend completely to the edges. Projector positioning is critical to eliminate the use of keystone projector adjustment, and to ensure the images meet exactly at the edges. The screen material is a polymer material stretched on an aluminum frame, manufactured to our specifications by Da-Lite, a world-leader in projection screens. The screen is free-standing and is assembled and leveled on-site using frames designed by NADS for this application.



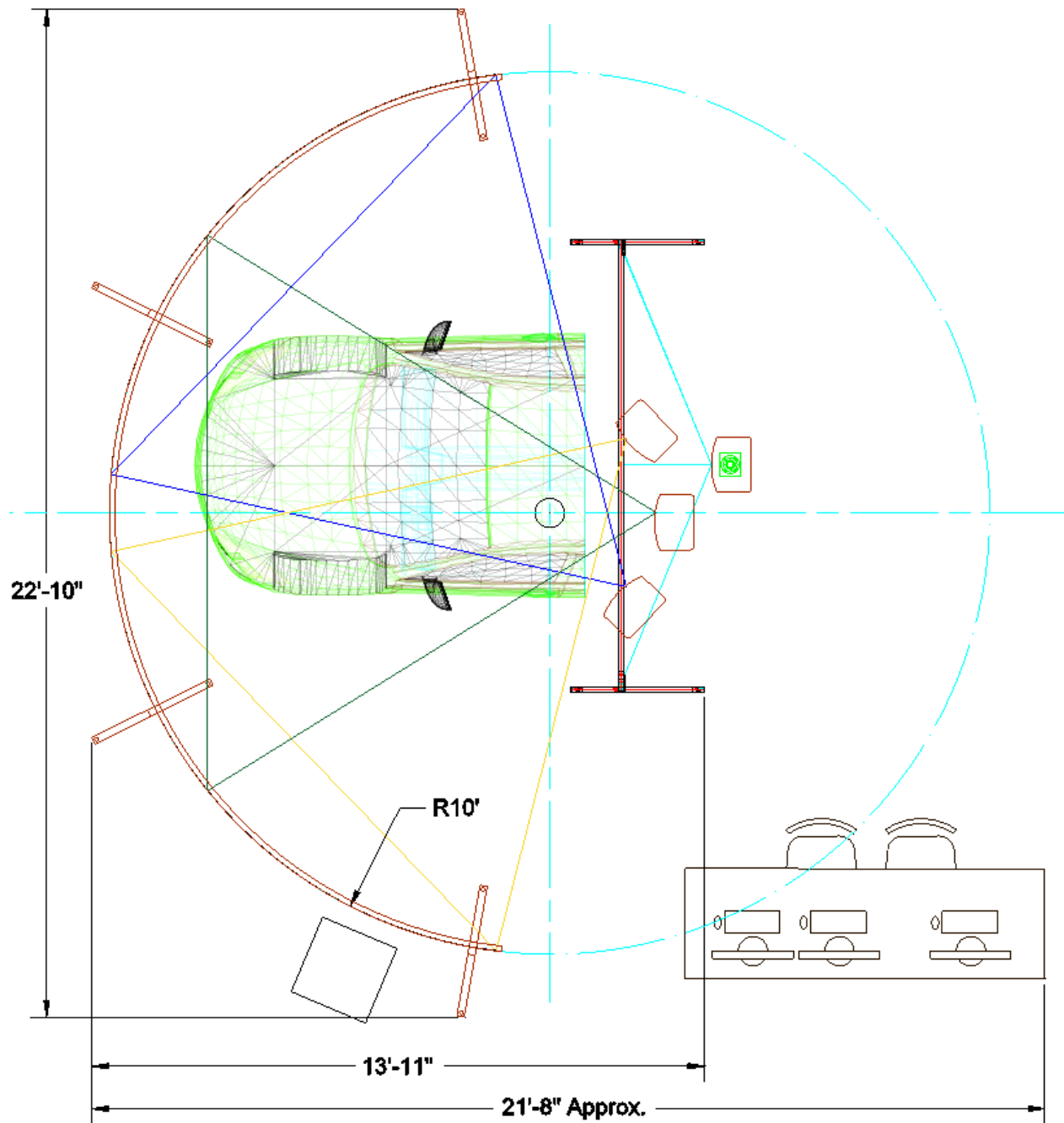
Plan View, Half-Cab with 170-Deg HFOV Flat Screens.



Triple Screen After Manual Alignment

Curved Screen

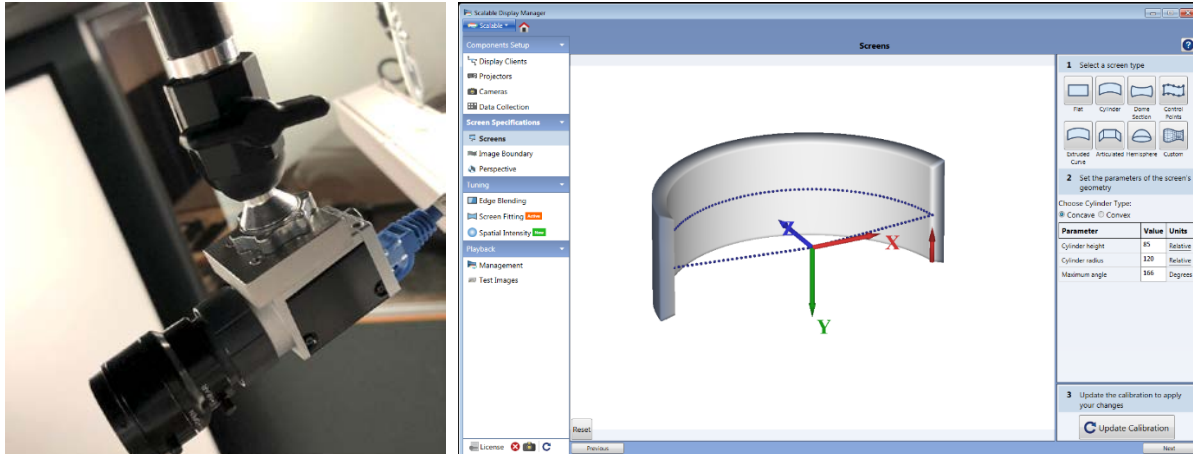
The curved screen is a polymer material stretched on an aluminum frame, manufactured to our specifications by Da-Lite, a world-leader in projection screens. The screen is free-standing and is assembled and leveled on-site. This screen has a matte black border and frame.



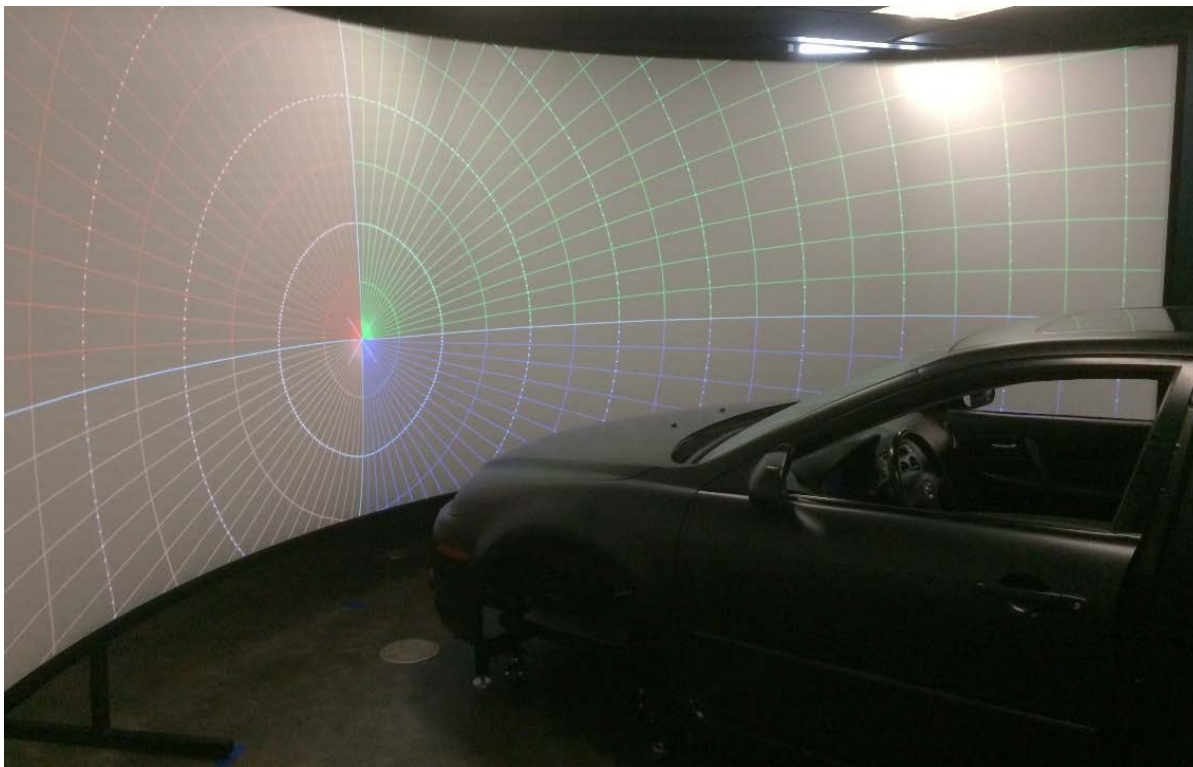
Plan View of Half-Cab miniSim with Curved Screen, 166-deg HFOV

Blending and Warping

The curved screen requires 3 projectors. These are positioned to fill the entire surface, and the overlapping areas are blended using software by Scalable Displays. This system, once properly configured, is automated and uses network cameras and an algorithm to generate the blend and a warp matrix which is applied automatically.



Network HD Camera and Warping and Blending System



Testing Projector Alignment on Curved Screen

Projector Mounting

The projection system and mounts are carefully designed for each application to ensure optimum performance and ease of installation. The locations for each projector are carefully calculated to ensure proper image geometry to eliminate undesired distortion.

During installation, the room is carefully surveyed to locate the screen and projector mounts exactly. Mounts and poles that allow precise roll, pitch, tilt, and height adjustments are used for projector alignment.

The mounts used will vary depending on the room. **A ceiling height of at least 9 feet (2.75m) is usually required.**

Dropped Ceilings

If there is a drop-ceiling, then it may be possible to use mounts that attach to the ceiling tile supports themselves. This is very straightforward if the desired locations do not conflict with the frames. For this to work well, the dropped ceiling **MUST** be properly installed and supported, and free of vibration (typically due to HVAC systems) that will affect image quality.



If the dropped ceiling is not strong enough or vibration is present, then the projector poles must be attached to rigid structures above the dropped ceiling.

Un-finished Ceilings

When no dropped ceiling is present, then the projectors can be mounted to existing structures, or a free-standing gantry can be used. The use of a plate preserves the relationship between the projectors precisely.



Half-Cab Power Requirements

The general power requirements are listed below.

Equipment Description	Rated Power [W]
Computer Rack (120Vac)	
MiniSim PC	800
Video Capture PC	500
LCD (ASUS 24")	35
LCD (ASUS 24")	35
Total:	1370
Cab (120Vac)	
Cab Power Supply (50A @ 13.8 Vdc)	750
Audio Amplifier (sub, tactile), 2x120 Wpc	250
Audio Amplifier (audio), 4x80 Wpc	175
LCD (12" instrument panel)	15
Misc	15
Total:	1205
Display System	
Short-Throw Projector	400
Short-Throw Projector	400
Short-Throw Projector	400
Ultra-Short-Throw Projector	400
Total:	1200



Computer Rack



Short-Throw Projector



Ultra-Short Throw Projector

NADS miniSim™ Software and PC

The **NADS Software** supplied with the miniSim™ simulator may consist of:

- NADS miniSim™ real-time driving simulation software
- NADS ISAT™ scenario authoring tool for creating/modifying scenarios
- NADS TMT™ virtual environment authoring tool with 92 sample tiles
- NADS nDaqTools™ and GUI for MATLAB® to assist in Data Reduction
- NADS AutoDriver™ vehicle automation system
- NADS Virtual World API
- NADS VidCap™ Video Capture Software
- NADS Infotainment System
- NADS Springfield Road Network

The typical miniSim PC hardware specification* is as follows:

- Rackmount or Tower Case
- Windows 10 64 bit Pro
- Intel Core i9 3.5GHz 10-Core Processor
- 2 SSD RAID (500 Gb ea, 500 Gb usable)
- 32 Gb DDR3 SDRAM
- NVIDIA **GeForce RTX 2080 Ultra** and **GTX 1660 Ultra** GPUs
- Matrox Triple Head to Go Digital SE

This hardware supports the following number of video channels and resolutions at **60Hz**:

- Instrument Panel: 16:9 LCD, up to 1920x1080, (1280x800, 1600x900 typical)
- Front: Left, Center, Right displays/projectors: 3 x 1600 x 900 (3 x 1280 x 1024 typ.)†
- One Auxiliary Video Output: up to 1920x1080 (1280x800, 1600x900 typical)
- Operator Display: 1920x1080

**Note: Hardware specs may change at any time due to hardware changes or availability.*

†Note: If full HD (1920x1080, 60Hz) is required, then NVIDIA Quadro P6000 and P2000 Cards are needed in lieu of the GeForce cards. Contact NADS for pricing.

Software Control and Keylok™

Two key miniSim applications are controlled via a USB security device called a Keylok. This device has a NADS-programmable expiration date, for which NADS provides renewals at the time of payment of the software support and maintenance fee.

miniSim and Tile Mosaic Tool (TMT) applications

Only two applications require a valid Keylok to operate:

- NADS miniSim™ real-time driving simulation software
- NADS TMT™ virtual environment authoring tool with 92 sample tiles

The miniSim and TMT applications are typically installed on the simulator PC. Two Keyloks are supplied so that, in addition to the simulator, the miniSim application can be run on a laptop or development PC for scenario testing and development.

The TMT is used to assemble road networks (or ‘map’) of your design. It is best run on the same machine where the miniSim installation is present, as the export process generates many files that are to be accessed by the miniSim when it runs a scenario.



Two Keyloks are supplied

Integrated Scenario Authoring Tool (ISAT) and nDAQTools

These applications do not require a Keylok to operate.

You may install ISAT on any Windows 7, 8, or 10 desktop or laptop so that you can **Create**, **Rehearse** and **Replay** scenarios anywhere. A single file created by the TMT is opened by ISAT to represent the map during scenario authoring.

The nDAQTools application runs within a MATLAB™ environment (not supplied), and is used to calculate measures from the time-history data recorded by the miniSim. This also can be done on any Windows PC with MATLAB installed. Some miniSim users have found that Python can be used instead of MATLAB, and have created an open source Python library called [unDAQTools](#) available on Github. The unDAQTools library is not supported by NADS, but we do supply a utility to convert the miniSim data files to text.

Software Description

The miniSim Driving Simulator contains the following core software components:

- Real-Time Core
- Simulation Control Front-End GUI
- Driving Control Feel and Instrumentation
- Vehicle Dynamics
- Scenario Control
- Visual Rendering
- Audio Engine
- Data Acquisition

Real-Time Core

The real-time core is the underlying engine that coordinates the execution of the various functionalities of the simulator and provides the real-time data communication mechanism among components, either through shared memory on the same computer, or through a local TCP/IP connection between different computers. The other components are built on top of the real-time core.

Simulation Control Front-End GUI

The front-end GUI provides control to the driving simulation process and to data collection and analysis. The operator can start and stop drives, select different scenarios, play presentation slides, and collect and analyze driving data through the front-end GUI. The GUI will be displayed on a small touchpad screen.

Driving Control Feel and Instrumentation

The driving control feel and instrumentation component reads the driver's control input via the steering wheel and foot pedals and passes it to the vehicle dynamics component. It also takes the vehicle data such as speed and engine RPM and renders them to the instruments, which are simulated on the main display.

Vehicle Dynamics

The vehicle dynamics component provides a physics-based vehicle model that simulates the vehicle the subjects drive during the runs. It is identical to the corresponding component in the large NADS simulators. By changing input data, the same software can be used to simulate different types of vehicles.

Scenario Control

The scenario control component is responsible for simulating the behavior of other vehicles and objects in the virtual environment. It is identical to the scenario control component in the large NADS simulators and takes the same scenario configuration files generated by ISAT as input.

Visual Rendering

The visual rendering component renders the visual scene to the main display(s) of the driving simulator. One PC with a high-performance graphics card will be used for the rendering of up to three main channels. The renderer is built on top of an OpenGL-based open-source graphics library.

A range of environmental conditions can be rendered in the environment using the scenario control system:

- Rain
- Snow
- Fog
- Wind (speed and direction)

The driver can control the windshield wipers. Precipitation accumulation, the effect of wind on the precipitation, and the wiping effect is simulated.

Audio Engine

The audio engine provides audio cues to the driver. The component produces sounds from the own vehicle, including engine, powertrain, and tires, as well as sounds from the surrounding environment. The PC that hosts the audio engine is equipped with a PC audio card and connected to a set of speakers.

Data Acquisition

The driver's data are collected at the same frequency that the miniSim runs. The set of data that can be collected includes driving control inputs, own vehicle states, and scenario data. All data are stored on file for replay and future analysis.



Figure 1 – Example of miniSim Weather Effects

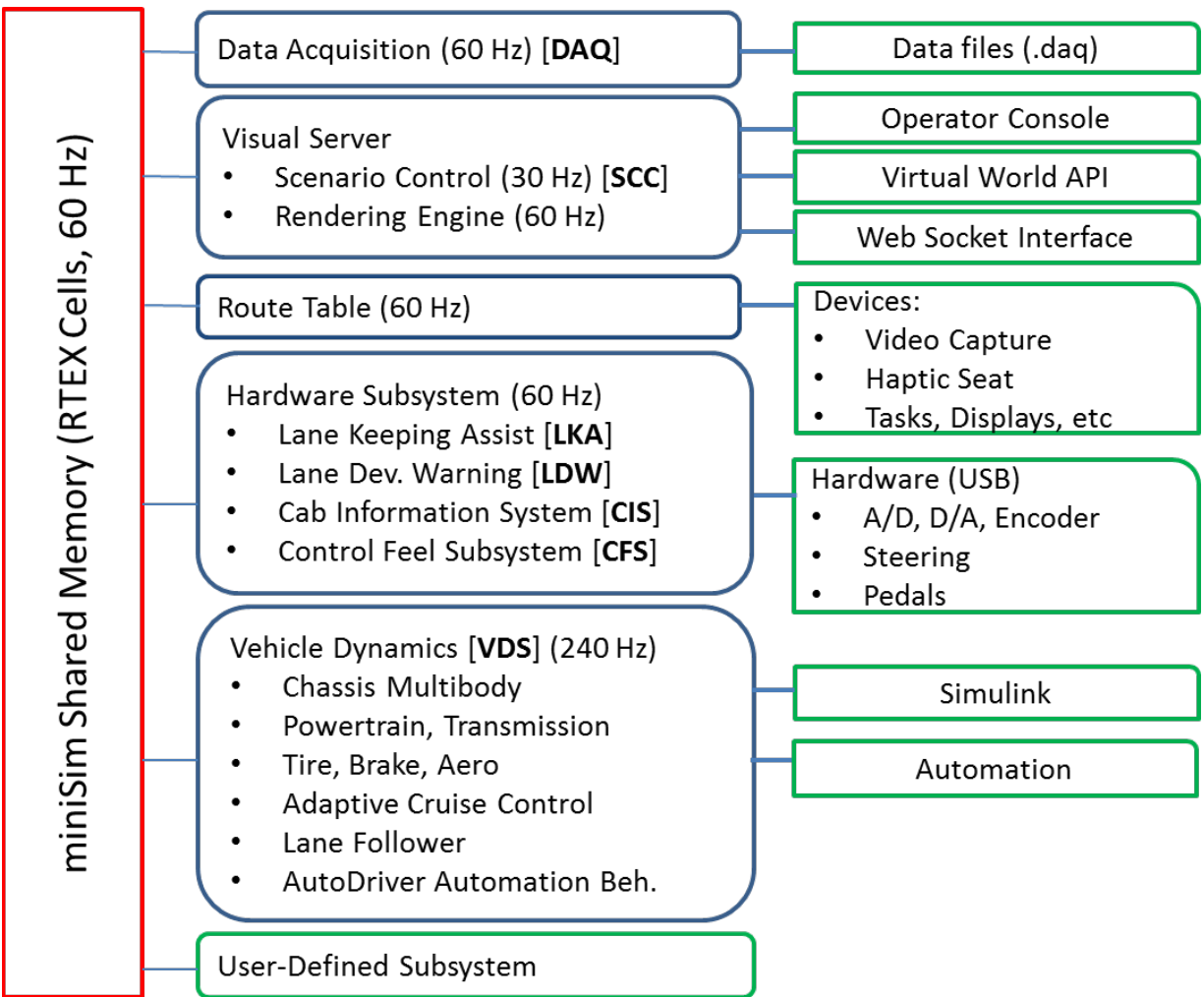


Figure 2 - miniSim(TM) computing architecture with optional User-Defined Subsystem

miniSim Operator Interface

The miniSim software interface is the primary method a user interacts with the simulator, Figure 16. The user interface has the following functionality:

Scenario Selection and Start/Stop

The user selects the scenario to run from a drop-down list and can start and stop the simulation.

Scenario Import and Delete

The user is able to import new scenarios and delete existing ones through a standard windows dialog.

miniSim™ Data Collection Mode

The miniSim data acquisition system (DAQ) runs every time a scenario is run. However, when the miniSim is placed in Data Collection Mode, the operator must select the Experiment Name and Test Subject ID from a list you create. The Experiment and Test Subject ID selected then restricts the allowable scenarios that can be run according to a matrix defined by the user – *ensuring that the data collection conforms to your experimental design*. In addition, you may enter notes or comments about a particular test subject or drive and these comments are saved for future reference, Figure 17. You may also manually enter The Experiment and Test Subject ID at runtime.

Replay Mode

The replay mode provides a method to document consistency of data across multiple simulators, an important consideration for multi-site data collections, for example. A driver input file is recorded, and the simulation re-run on multiple simulators and the data compared. The driver input file is specific to the road network and scenario being driven. The Runtime Measures Evaluation, described below, offers a quick and easy method to compare and document performance across simulators.

Runtime Measures Evaluation

While the miniSim data collection system (DAQ) collects well over 100 variables for post-processing, the Runtime Measures Evaluation functionality offers the researcher the option of obtaining measures from the simulator immediately when the drive ends. The measures are reported for the entire drive, and for up to 20 events. The user has total control in defining the start and end points of each event through the scenario.

The current list of runtime measures are as follows:

Collision Count	Lane Departure Count
Maximum Speed	Lane Departure Percentage
Minimum Speed	Speeding Count
Average Speed	Speeding Percentage
Standard Deviation of Speed	Average Headway
Standard Deviation of Lane Position	

Other measures are possible – just ask us!

Driver Input Calibration

The miniSim has two easy-to use features that allow calibration of the driver inputs. This is important, as many driver input mechanisms use analog sensors (potentiometers) for steering wheel and pedal position, and differences between setups and controls are possible and must be accounted for.

One feature is an automatic calibration routine that asks the operator to turn the wheel in each direction to the end stop and press on each of the pedals. The miniSim captures this data, and

produces a calibration file. The second feature is a verification of the driver inputs – the operator can operate the controls and verify that the calibration values are being applied correctly.

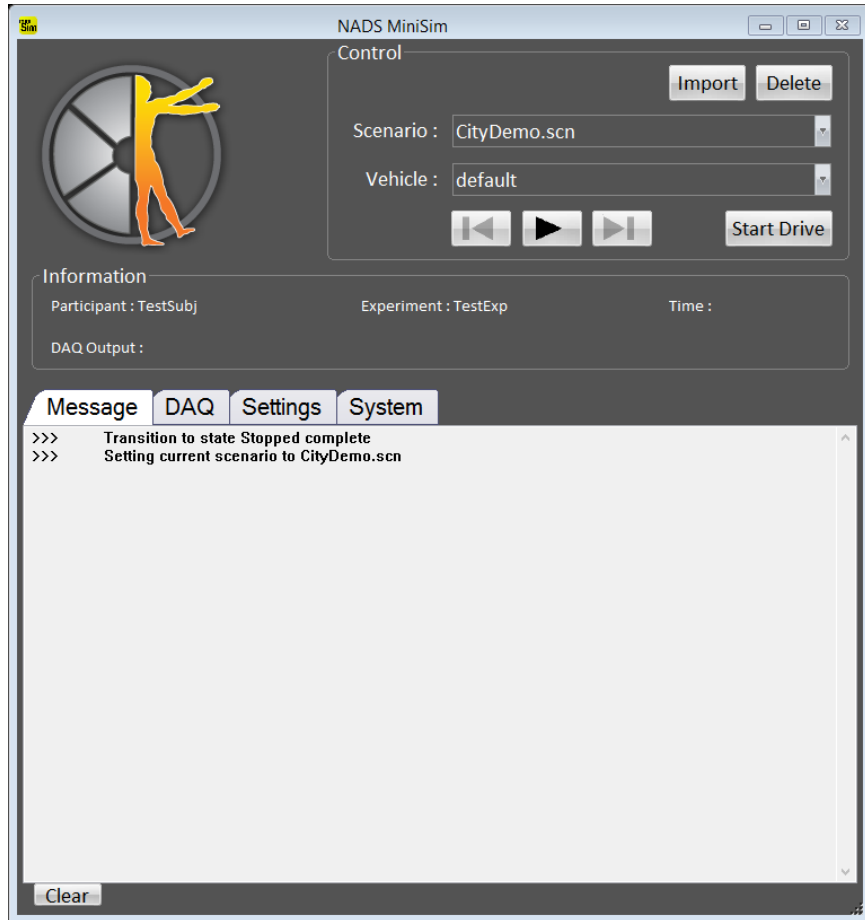


Figure 3: miniSim User Interface

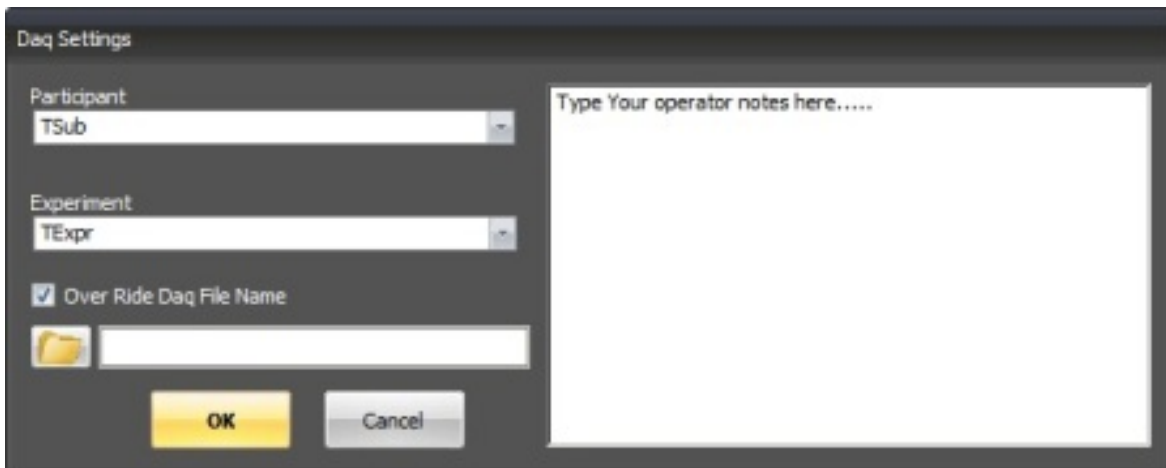


Figure 4: miniSim Data Collection Window

Dependent Measures

Values and locations of Maximum and/or Minimum

Typical values are maximum brake pedal force, maximum steering wheel angle, maximum/minimum longitudinal/lateral acceleration and maximum/minimum own vehicle speed. These values are usually used to study the severity of the driver's action or movement, particularly useful in examining stopping or slowing in unsafe circumstances or unsafe intersection behavior typical of impaired drivers.

- Time to Maximum Brake Press: Time from the start of the event until the driver has maximum brake displacement
- Maximum Deceleration
- Maximum Steering Rate
- Maximum Lateral Acceleration
- Maximum Lateral Jerk

Average and deviation

Values can be used to study the driver's behavior while distracted or impaired. For example, the lane deviation of the driver's vehicle has been used to study the influence of alcohol on the driver. These values better capture the variation of driver's performance versus more extreme responses captured in minimum/maximum values.

- Average own vehicle speed
- Lane deviation of driver vehicle
- Average brake pedal force
- Average steering wheel angle.

Reaction time

Values include accelerator pedal release time, brake pedal press time, sign-recognition reaction time, etc. These values are used to study the driver's reaction time during an event. For example, if the driver is asked to press a button upon seeing a certain sign, the time between the sign appearing and the button press is the sign-recognition reaction time. This measure indicates how fast and well a driver is monitoring the environment, particularly relevant for landmark and traffic sign identification tasks, other tasks designed to assess visual search and target recognition, or route following tasks using verbal directions to a destination to test verbal memory, visual perception and attention.

- Accelerator Release Reaction Time: Time from the start of the event until the driver has released the accelerator pedal
- Brake Press Reaction Time: Time from the start of the event until the driver initially presses the brake pedal
- Button-push or Verbal Response Reaction Time

Steering Responses

Erratic steering, leading to shoulder incursions or lane excursions, are often indicators of increased driver workload.

- Time to Steering Onset: Time from start of the event until the driver has a corrective steering input exceeding 5 degrees
- Length of Excursion
- Extent of Excursion
- Area of Excursion

- Duration of Excursion

Lane change detection

This measure includes the detection of a lane change, how long the lane change takes, lane change angle, etc. Lane changes are very common events in real life.

- Minimum Time-To-Line Crossing: The minimum time-to-collision from event onset until end of the event.
- Time-To-Line Crossing At Accelerator Release
- Time-To-Line Crossing At Brake Press
- Time-To-Line Crossing At Steering Onset

Eye-tracking (Optional)

Eye-tracking is used to study the driver's eye movements during the events. Measures include time spent looking forward, into mirrors or over the shoulder. These measures can be used to examine how the driver observes the environment generally, as well as for specific tasks measuring attention, divided attention, and selective attention while driving.

- Eyes on Road Time
- Eyes on Target Time
- Eyes off road at event onset: An indication of distraction

Crash detection

Crash detection includes detecting the crash, the speed of the driver vehicle at the time of the crash, and the speed of the other vehicle at the time of the crash (if applicable). *Conflict risk*. Conflict risk is also used to study the degree of danger between the own vehicle and a lead or adjacent vehicle. The degree of danger depends on the speed of the two vehicles and the distance between them.

- Collision: Indicates whether the driver's vehicle collided with the braking lead vehicle
- Relative Velocity at Collision: Indicates the difference in speed at the time the vehicles collided.
- Near Misses: Indicates that the driver was near colliding with the lead vehicle.
- Conflict: Indicates that the driver was in conflict with the lead vehicle.

Time to collision

Time to collision is defined as how long it would take the own vehicle to collide with another object if the own vehicle maintains its current status.

- Minimum Time-To-Collision: The minimum time-to-collision from event onset until end of the event
- Time-To-Collision at Accelerator Release
- Time-To-Collision at Brake Press
- Time-To-Collision at Maximum Brake
- Adjusted Minimum TTC: Minimum Time-to-Collision adjusted to account for crashes. Negative values indicate how much sooner the driver would need to have responded to avoid the collision.

Data Stream Contents

A large collection of variables are collected at during a person's drive on the miniSim. At the end of the drive, a DAQ file is written with a large number of variables that log the state of the simulation at each moment during the person's drive. A DAQ file can be loaded and viewed in the ISAT™ which allows users to playback the drive and view/graph the state of each variable. A free utility called **daqConvert** is also included that allows users to convert a DAQ file into Matlab or text/ASCII formats. Another free utility called **daqViewer** is also included that provides a quick way to plot variables from a DAQ file.

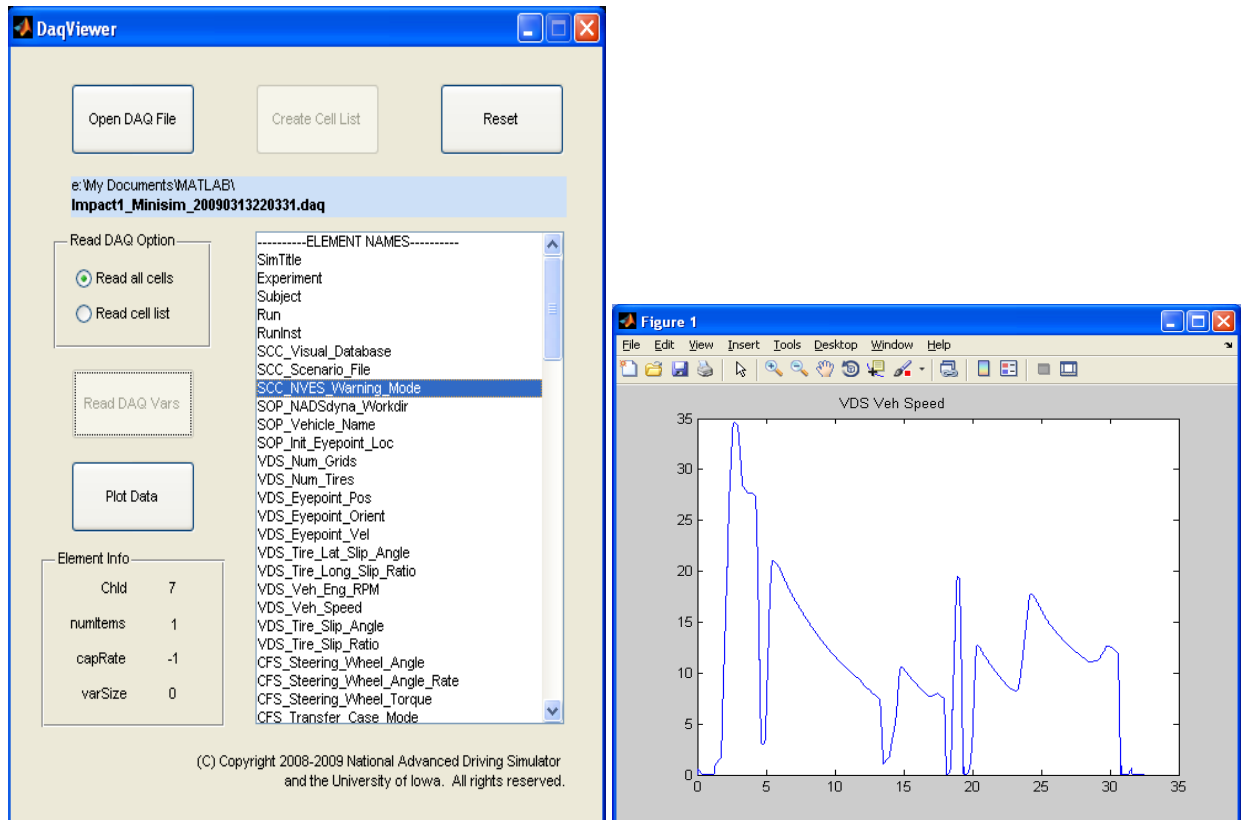


Figure 5 - daqViewer Screenshots (MATLAB GUI)

A partial listing of the data acquisition variables are shown below:

Driver Input Variables

These variables provide information about the driver's inputs to the driving simulator. These include information from the steering wheel, accelerator, brake, shifter and turn signal inputs.

Driver Input Variables	Definition and Units	Collection Frequency
Accelerator Pedal Position	Normalized value between 1 and 0 indicating the accelerator's position. A value of 0 implies that the accelerator is fully released and value of 1 implies that the accelerator is fully depressed.	60 Hz
Auto Transmission Mode	Indicates the current gear that the transmission is in. -2 = Park, -1 = Reverse, 0 = Neutral, 1 = First, 2 = Second, 3 = Drive, 4 = Overdrive	Differential
Steering Wheel Angle	The steering wheel's angle (in degrees). Values typically range between -202.5 to +202.5 degrees. Negative values indicate that the steering wheel is being turned to the left whereas positive values indicate that the steering wheel is being turned to the right.	60 Hz
Cruise Control State	The state of the cruise control button 0 = Not available, 1 = off, 2 = On, 3 = Set/Accel, 4 = Resume, 5 = Coast	Differential
Car Horn	1 – off, 2 - on	Differential
Turn Signals	Indicates the state of the turn signal driver input 1 = no turn signal on, 2 = left turn signal on, 3 = right turn signal on	Differential

Collision Detection Variables

These variables provide information about which objects the driver has collided with during their drive.

Collision Detection Variables	Definition and Units	Collection Frequency
Collision Count	How many collisions with the ownship vehicle have taken place so far. The value ranges between 0 and n where n is the total of number of collisions in the entire drive so far.	60 Hz
Collision List Size	Number of objects that are colliding with the driver's vehicle and are present in the collision list. Values range between 0 and 10. Note: A maximum of 10 collisions are recorded per frame of execution. Priority is given to objects that are on the list in the previous frame.	60 Hz
Collision Object SOL Identifiers	The SOL identifiers of objects that are colliding with the driver's vehicle. The SOL identifier gives the user an indication of what type of object has collided with the driver. This is an array of 10 variables. If the value of the Collision_List_Size is n, only the first n values in this array are valid. Each valid value ranges between 1 and n where n is the highest SOL identifier.	60 Hz
Collision Object Type Identifiers	The type identifier of the objects that are colliding with the driver's vehicle. This is an array of 10 variables. If the value of the Collision_List_Size is n, only the first n values in this array are valid. Valid values are: 1 – trajectory follower (DDOs) 2 – vehicle (can be ADOs or static objs) 7 – traffic signs 9 – obstacle 13 – walker	60 Hz
Collision Object CVED Identifiers	The CVED identifiers of the objects that are colliding with the driver's vehicle. A CVED identifier uniquely identifies the exact object that is colliding with the driver's vehicle. This is an array of 10 variables. If the value of the Collision_List_Size is n, only the first n values in this array are valid. Each valid value ranges between 2 and n where n is the highest CVED identifier.	60 Hz

Scenario Variables

These variables provide information about the state of the virtual environment and the dynamic objects (other traffic) around the driver.

Scenario Variables	Definition and Units	Collection Frequency
DynObj Data Size	<p>An integer that indicates the number of scenario objects around the driver's vehicle that information is being logged for.</p> <p>The maximum number of objects which data can be logged is 20.</p>	60 Hz
DynObj SOL Identifiers	<p>The SOL identifiers of scenario objects around the driver's vehicle. The SOL identifier identifies the dynamic object from the SOL library. This is an array of 20 integers</p> <p>If the value of the DataSize is n, only the first n values in this array are valid. Each valid value ranges between 1 and n where n is the highest SOL identifier.</p>	60 Hz
DynObj CVED Identifiers	<p>The CVED identifiers of scenario objects around the driver's vehicle. The CVED identifier uniquely identifies an object in the entire simulation. This is an array of 20 integers.</p> <p>If the value of the DataSize is n, only the first n values in this array are valid. Each valid value ranges between 1 and n where n is the highest SOL identifier.</p>	60 Hz
DynObj Headings	<p>Headings (in degrees) of scenario objects around the driver's vehicle. This is an array of 20 floats and objects are sorted with their distance to the driver's vehicle.</p>	60 Hz
DynObj Names	<p>Names of scenario objects around the driver's vehicle. The user may assign names to each object using the ISAT™. This is an array of 20 char arrays. Objects are sorted with their distance to the driver's vehicle.</p> <p>If the value of the DataSize is n, only the first n values in this array are valid. Each valid value ranges between 1 and n where n is the highest SOL identifier.</p>	60 Hz
DynObj Positions	<p>Positions (x,y,z in scenario coordinates) of scenario objects around the driver's vehicle. This is an array of 60 floats. Objects are sorted with their distance to the driver's vehicle.</p>	60 Hz

	<p>If the value of the DataSize is n, only the first n values in this array are valid. Each valid value ranges between 1 and n where n is the highest SOL identifier.</p>	
DynObj Roll and Pitch	<p>Roll and pitch values (i,j,k) of scenario objects around the driver's vehicle. This is an array of 120 floats. Objects are sorted with their distance to the driver's vehicle.</p>	60 Hz
	<p>If the value of the DataSize is n, only the first n values in this array are valid. Each valid value ranges between 1 and n where n is the highest SOL identifier.</p>	
DynObj Audio and Visual States	<p>Indicates the current audio and visual settings for the scenario objects around the driver's vehicle. This is an array of 40 integers. Objects are sorted with their distance to the driver's vehicle.</p>	60 Hz
	<p>If the value of the DataSize is n, only the first n values in this array are valid. Each valid value ranges between 1 and n where n is the highest SOL identifier.</p>	
DynObj Velocity	<p>Velocities (in ft/s) of the scenario objects around the driver's vehicle. This is an array of 20 floats and objects are sorted with their distance to the driver.</p>	60 Hz
Lead Vehicle Information	<p>This array provides information about the vehicle in front of the driver. This an array of 9 floats with the following elements:</p> <p>1st – CVED identifier of object (-1 = no vehicle in front of driver)</p> <p>2nd - distance to lead vehicle (in feet)</p> <p>3rd - bumper-to-bumper time to lead vehicle (in seconds)</p> <p>4th - bumper-to-bumper distance to lead vehicle (in feet)</p> <p>5th - time-to-collision (in seconds)</p> <p>6th - lead vehicle velocity (ft/s)</p> <p>7th – x coordinate of lead vehicle</p> <p>8th – y coordinate of lead vehicle</p> <p>9th – z coordinate of lead vehicle</p>	60 Hz
Lane Deviation	<p>Information about the driver's vehicle and it's current lane position. This is array of 4 floats:</p>	60 Hz

1 st : -1 (on an intersection corridor), 1 (on a lane)
2 nd : lateral offset from the center of lane/corridor
3 rd : width of the current lane (corridor's width is not reported)
4 th : Lane/corridor CVED identifier

Vehicle Dynamics Variables

These variables provide information about the state of the driver's vehicle during the drive.

Vehicle Dynamics Variables	Definition and Units	Collection Frequency
Surface Tire Index	0=Intersections and drivable off-road, 14=Road, 20=Shoulder	Differential
Chassis CG Acceleration	The driver vehicle's acceleration in ft/seconds ²	60 Hz
Chassis CG angular velocity	The driver vehicle's angular velocity in degrees/seconds	60 Hz
Chassis CG Orientation	The driver vehicle's orientation in degrees	60 Hz
Chassis CG Position	The driver vehicle's position (in feet)	60 Hz
Chassis CG Velocity	The driver vehicle velocity (in mph)	60 Hz
Eye-point Orientation	The driver's eye-point orientation (in degrees) in global coordinates	60 Hz
Eye-point Position	The driver's eye-point position (in degrees) in global coordinates	60 Hz
Head-point Angular Velocity	The head-point's angular velocity (in degrees/second)	60 Hz
Head-point Specific Forces	Forces being felt at the head-point (in G's)	60 Hz
Load Torque	Wheel torque due to external forces (in foot lbs.)	60 Hz
Num Grids	Number of grids used for each contact patch	Differential
Num Tires	Number of tires on the vehicle (0 -10)	60 Hz
Steering Torque Backdrive	Commanded steering wheel torque (in foot-lbs.)	60 Hz
Tire Ground Contact	The tire/terrain contact location (in feet)	60 Hz
Tire Rotational Velocity	Tire rotational velocity (in degrees/second)	60 Hz
Tire Slip Angle	Tire slip angle (in degrees)	60 Hz
Tire Slip Ratio	Tire slip ratio (0.0 – 1.0 normalized)	60 Hz
Tire Weight on Wheels	Tire weight on wheels (in lb. force)	60 Hz

Vehicle Engine RPM	Vehicle engine revolutions per minute	60 Hz
Engine Torque	Vehicle engine torque (in foot-pounds)	60 Hz
Vehicle Heading	Vehicle heading (in degrees)	60 Hz
Vehicle Speed	Vehicle speed (in mph)	60 Hz
Vehicle Transmission RPM	Vehicle transmission revolutions per minute	60 Hz
Vibration Forces	Commanded vibration forces (in G's)	60 Hz
Vehicle Center Heading	Heading angle of wheel (in degrees)	60 Hz
Wheel Center Velocity	Translational velocity of wheel center (in feet/seconds)	60 Hz
Wheel Spin	Wheel spin (in radians/second)	60 Hz
Wheel Spin Angle	Rotational position of tire (in radians)	60 Hz
Wheel Steer Angle	Road wheel angle (in radians)	60 Hz

ISAT™ (Interactive Scenario Authoring Tool)

The NADS ISAT™ is one of the most full-featured scenario authoring tools available anywhere in the world. This tool, originally built in 1996, is currently in its 4th major revision. For the last 11 years, this tool has been successfully used for both research in the NADS family of simulators and training in the Swiss FATRAN Military Truck Driving Simulator.

The ISAT™ provides a user-friendly graphical interface for developing training scenarios. ISAT features include the following:

- Easy-to-use fully GUI interface, no programming or script writing is required
- Robust control set which allows the creation of training scenarios with a high-level of repeatability and control
- A.I. (artificial intelligence) driven debugging mode for quick testing and off-line perfection of scenarios
- Graphical playback of data files collected from drives of the miniSim™
- Production of media content, including pictures and videos
- Compatibility with the NHTSA scenarios

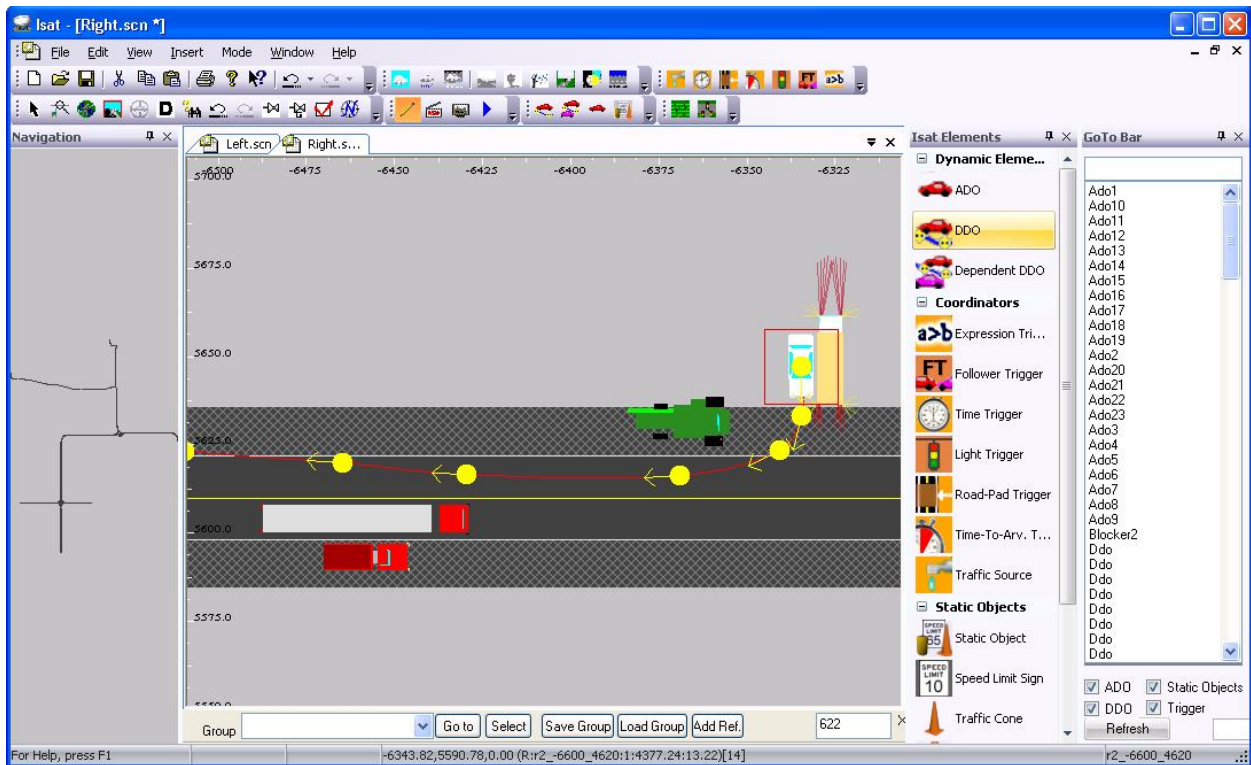


Figure 1 - NADS ISAT(TM) scenario authoring tool

Please refer to the ISAT™ Users Guide for more information about the countless features of this powerful tool.

TMT™ (Tile Mosaic Tool)

The National Advanced Driving Simulator relies upon the TMT software for construction of correlated synthetic simulation environments. The software and graphics tile libraries are supported and maintained by a team of staff with over 40 years of combined simulation experience.

The TMT uses a modular, extendable library of visual and correlated logical or virtual components. A library of approximately 50 tiles will be included. Each tile is constructed as a self-contained set of files that function as a single entity. Each file set contains visual features, and may be populated with some or all of the following: roads, intersections, signs, collision elements, terrain objects and location markers called placeholders. The set of files is commonly referred to as a singular entity called a Database Tile. The library of modular component Tiles is called a Tile Library. The Tile Mosaic Tool (TMT) is an easy to use, graphical user interface application used to assemble previously created terrain and correlated databases, and publish them into a simulation environment. Simulation rendering performance can be adjusted through the TMT – there is no need to re-work source Tile database files.

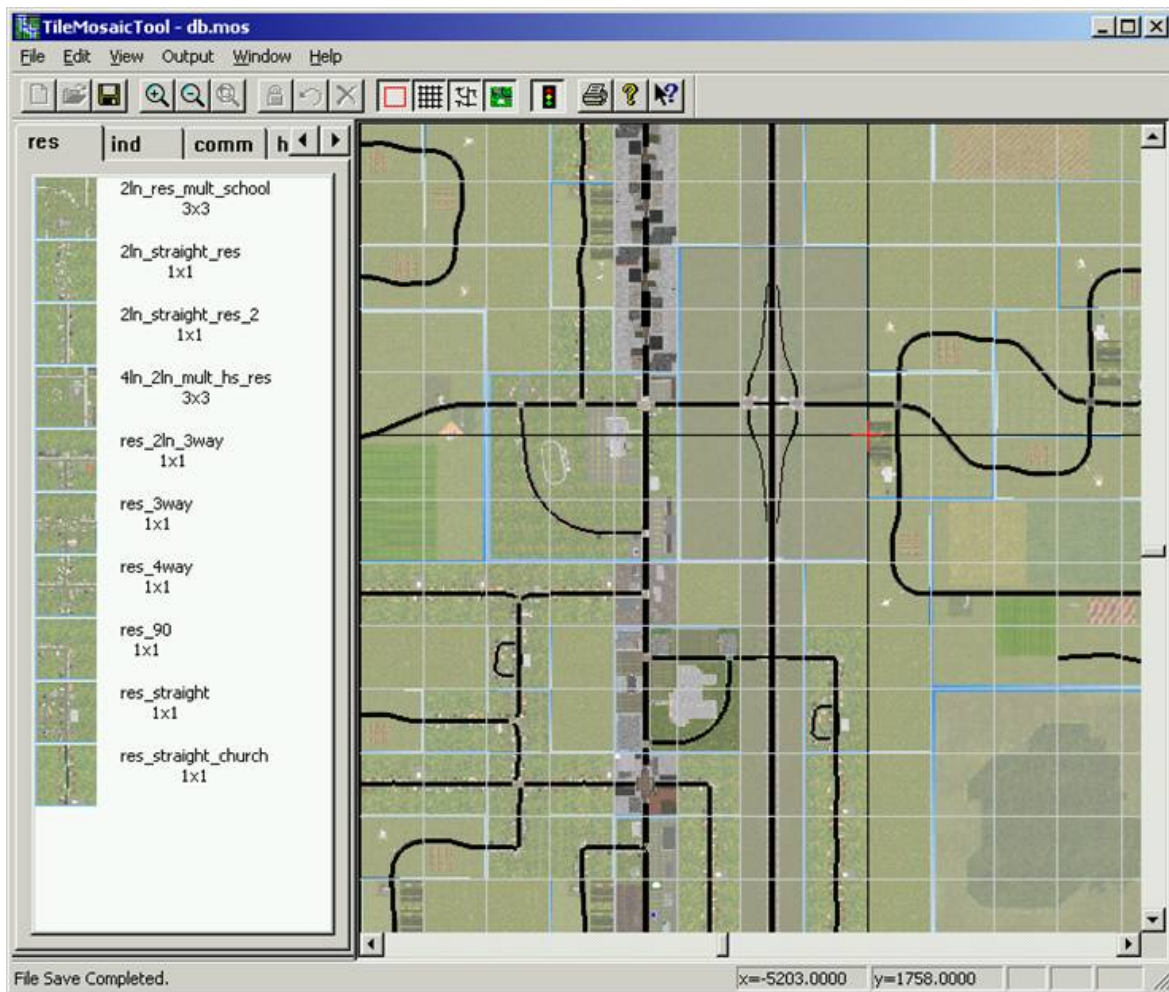
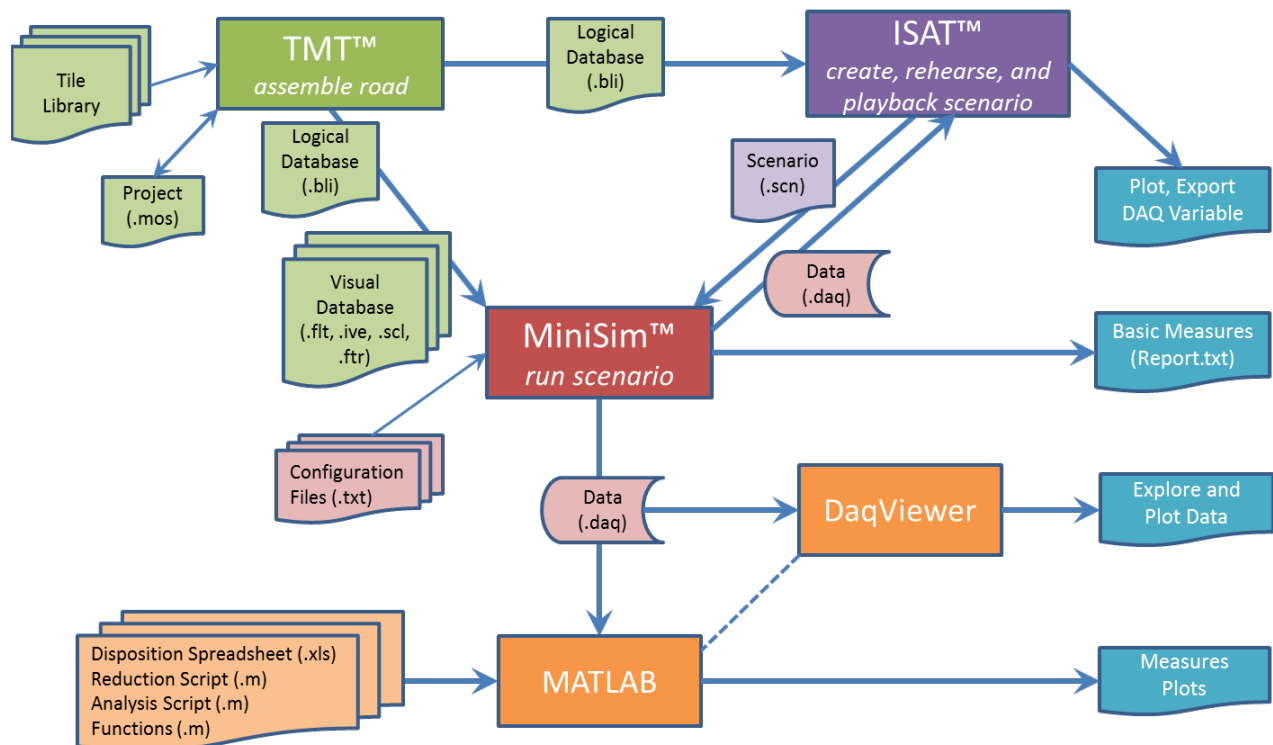


Figure 1 - Geo-typical Tile database assembly within the Tile Mosaic Tool

Relationship Between Software Modules

The following diagram shows the relationship between the functionality provided by the miniSim, ISAT, TMT and Matlab. In general, the workflow is as follows:

- The Tile Mosaic Tool (TMT) is used to assemble a road network out of pre-existing tile assets. The TMT creates a visual database and a logical database (the road network) that are used by the miniSim.
- The Interactive Scenario Authoring Tool (ISAT) is used to create a scenario that is specific to the road network. The ISAT can also play back a miniSim data file and display and plot data for scenario testing and debugging purposes.
- The miniSim driving simulator then runs the scenario on the road network using the driver's inputs and produces a time-history data acquisition file containing all the simulator variables sampled at 60 Hz.
- To reduce the time-history data into final measures, Matlab is typically used along with NADS-supplied graphical user interfaces and tools.
- The miniSim will also produce basic driver performance measures in text format, with no post processing required.



Vehicle Automation in miniSim™

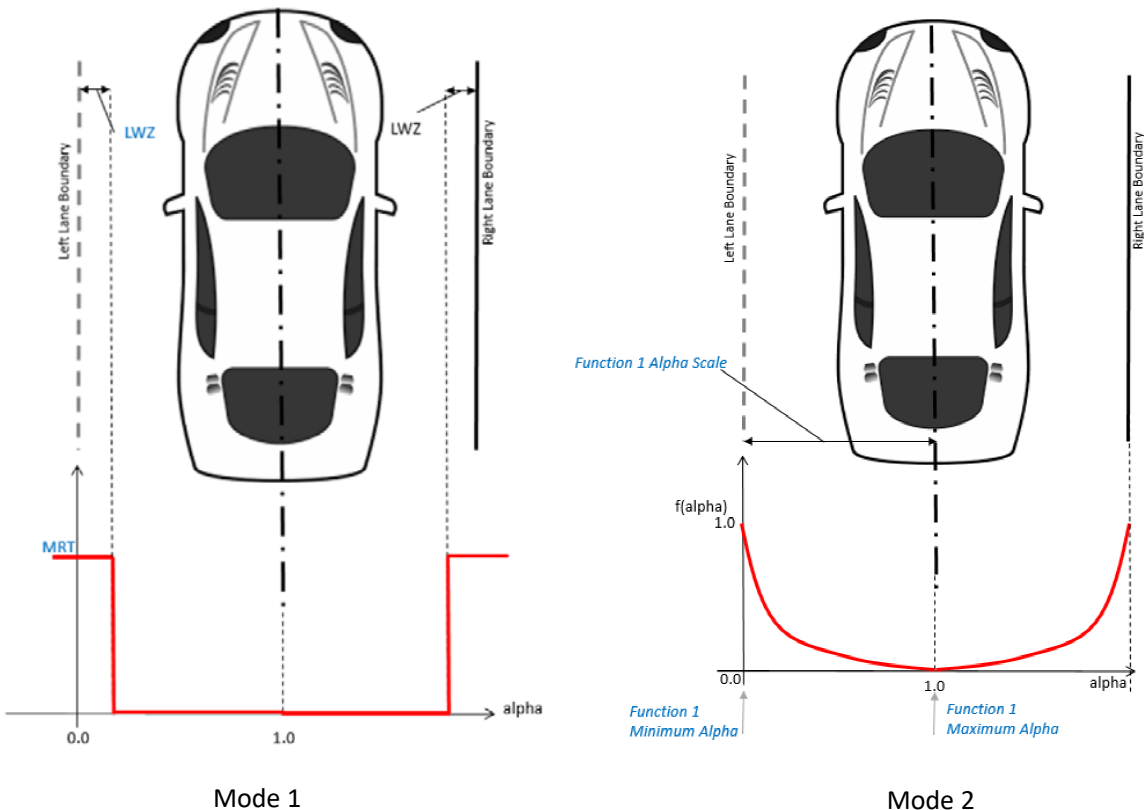
Built-In Driver Assistance Systems

All miniSim simulators, regardless of hardware have the following functions available that can be controlled via scenario or by the driver:

- Adaptive Cruise Control (ACC)
 - Normal or Full-Range operation.
 - All parameters are adjustable
 - Does *not* have knowledge of speed limits (see NADS AutoDriver™ for this function)
- Lane Following
 - Automatically steers down the lane
 - Does *not* adjust vehicle speed for turns (see NADS AutoDriver™ for this function)
 - Steering wheel turns (Fanatec or NADS loader only)
- Lane Keeping Assist (LKA)
 - Provides torque cues when nearing the lane boundary
 - Turn signal suppresses cues

By the use of Lane Following and ACC together a simple autopilot function can be achieved. Using existing scenario triggers and actions, visual and audio alerts and system failures can be simulated.

The Lane Keeping Assist function has two torque modes, where the applied torque is either a step function or a normalized exponential function, where MRT is the Maximum Relative Torque.



Advanced Automation Support Option

This option requires a one-time fee of \$3500 and provides support for the advanced automation options available in the miniSim, consisting of:

- Virtual World API
- NADS AutoDriver™ Vehicle Automation
- Hardware Subsystem

Virtual World API and Example Program

This Logical Database API is to provide a simplified programmer interface to serve as a method to extract road and scenario-object information at run time on the MiniSim, when provided with the location of the driver's own-vehicle (i.e. x,y,z). While this API will not directly serve as an interface into the MiniSim, it will however, be able to open the BLI file and run parallel with the MiniSim.

As part of the delivery an example program that does provide a communication interface to the MiniSim, and utilizes this API will be delivered. The below figure shows the data-flow, where the example program receives the current state information including the position of the driver, the example program then uses this information to query the API.

The API is developed in C++; it will be delivered as a library file built in VS 2010, with a header files. The API is object oriented in design and will use Hungarian notation (classes begin with C, member variable will begin with m_). The example program will be provided as a source code distribution, with a project and solution file for VS 2010. This program will be developed with C++. The operation of the example program will require network connectivity with the MiniSim. The API will require in the installation directory of the executable:

- a copy of the BLI
- a copy of the SOL file
- a copy of the scenario file

Definitions:

CVED:	Correlated Virtual Environment Database is an API used internally at NADS to read the bli, and query the road network.
BLI:	Binary Logical road Interface, this is a file that contains all of the logical road information.
SOL:	The sol file provides static information about static or dynamic objects, such as the bounding box size
Corridor:	A corridor is a defined path through an intersection
Static Object:	Static objects include signs, and non-moving objects that are either part of the BLI, or inserted into ISAT by the scenario author.
Dynamic Object:	Dynamic objects are those objects that when created by running the scenario, have a physical presence in the simulation, and are updated by scenario control. These are either ADOs or DDOs.

Available API Calls:

Road Information (for a given XYZ location)	
GetLaneWidth()	Returns width of the current lane
GetRoadMarkings()	Returns lane marking types (both sides) and also distance from center line to inside of lane markings (*see Note)
GetLaneLayout()	Returns lane count, direction of each lane, and meta-data about the lane
GetLaneSurfaceType()	Returns road surface type; documentation provided for type codes
GetRoadTrace()	Returns an array of x,y,z points and corresponding lane width (trace), following the road. Different versions will: <ul style="list-style-type: none"> • Use the driver's scenario path to navigate intersections that the function encounters. • Take a direction (right, left, straight), and navigate intersections • Take an intersection corridor id, and provide a trace through the corridor through the intersection
GetOncomingIntersection()	Returns the distance to the next intersection. If currently located on an intersection, it returns a list of connected corridors, and their respective turn directions.
GetOncomingHaltLine()	Returns the distance and x,y,z location of the next halt line. Also will take the driver's path to determine next halt line.
GetOncomingMergePoints()	Returns distances to and x,y,z locations of points where two corridors begin to overlap
GetLightState()	Returns traffic signal state
GetCurvature()	Returns the radius of curvature
GetGrade()	Returns road grade in degrees
ChangeLanes()	Returns distance to indicated lane or a code indicating if the lane does not exist, or is an oncoming lane.
Static Object Functions	
GetStaticObjects()	Given a bounding box, this returns a list of static objects with their x,y,z location and SOL ID.
Dynamic Object Functions	
UpdateDynamicObjects()	Takes in dynamic object data from the MiniSim, and adds entries into CVED for each object.
GetDynamicObjects()	Given a bounding box, this function returns a list of dynamic objects with their x,y,z location and SOL ID.
GetTerrainObjects()	Given a bounding box, this function returns a list of terrain objects that have been intentionally placed in the BLI. This tends to only be a much smaller subset of what is in the visual database and is used for things like parking lots and buildings.
GetSolInfo ()	Given SOL ID, this returns information from the SOL file, including the minimum bounding box for the object.

**Note: Only a minority of tiles have tags (or attributes) that allow road marking information to be queried by CVED, hence this API will only provide road marking information from BLIs marked with tags for road markings. Should a road position not have valid markings tag, a return code will indicate no information was available.*

Virtual World API Example program

- An example program is provided, this program:
- Every query function accessible to the API is called at least once
- Receives required data from miniSim., and properly loads the data into the API's objects.
- The source code and project files are provided, compatible with Visual Studio 2010.

NADS AutoDriver™ Vehicle Automation

An automated driving capability has been added to the NADSDyna vehicle dynamics software to take over operation of the steer, gas and brake controls to autonomously follow a route through the world. The behavior characteristics are controlled through a set of parameter files and a command vocabulary that may be given through the following mechanisms:

Scenario action (ie triggered by scenario event)

Manual action (ie driver button press)

External input, typically via UDP

There are functions for transfer of control, high-level driving style parameters, and support for existing scenario control triggering systems. Along with the logical database API, it provides the user with the tools to develop their own high-level systems for executive control. The functions are as follows:

- Lane Following
- Speed Following
- Lane Change
- Brake to Stop
- Merge / Exit
- Turning
- Pull Over
- Abort
- Transfer of Control
- Driving Style
- *Triggers*

The driverParams.txt file has parameters for the NADS auto-driver model. This model makes use of the concept of driver states and driver style. Driver states indicate what the vehicle is doing; and driver style indicates how it carries out its actions. The current driver states are: default driving, braking to a line, free braking, turning, changing lanes, and pulling over. Each parameter is provided with two limits. The driver style is a continuous setting between 0 and 1 that varies all other parameters between their two limits. Generally, the extremes of the styles can be thought of as conservative and aggressive.

Handling characteristics of vehicle models are strongly dependent on speed, in fact on speed squared. For this reason, separate tables were created for the proportional and derivative steering PID parameters. In most cases, the steering controller will read these two parameters from the velocity-dependent tables rather than the driverparams.txt file. The exception to this rule is the pulling over driving state which does not use the tables. This approach of varying the parameters according to some other variable is called gain scheduling.

The turn-by-turn navigation instructions generally could be entered from the keyboard, programmed into scenario triggers, or transmitted over a network connection from a remote subsystem. The approach has been to define 'hot key' commands that consist of single characters. This system hopefully makes the various commands easier to remember.

Table 1. NADS Auto-driver Parameters

Parameter	Description	Notes
steer_kp	Proportional gain applied to lane offset error	overridden by gain scheduling
steer_kd	Derivative gain applied to look-ahead offset error	overridden by gain scheduling
steer_ki	Integral gain applied to integrated offset error	
steer_min	Minimum steering angle	(deg)
steer_max	Maximum steering angle	(deg)
lookahead_time	Time headway to use for look-ahead offset point	(sec)
lookaheadrate_lim	Rate limit for changing the look-ahead time	
throttle_kp	Proportional gain applied to velocity error	
throttle_kd	Derivative gain applied to acceleration	
throttle_ki	Integral gain applied to integrated velocity error	
throttle_fwd	Forward throttle signal gain	
throttle_min	Minimum throttle command	(0-1)
throttle_max	Maximum throttle command	(0-1)
brake_kp	Proportional gain applied to velocity error	
brake_kd	Derivative gain applied to acceleration	
brake_ki	Integral error applied to integrated velocity error	
brake_fwd	Forward brake signal gain	
brake_min	Minimum brake command	(lbf)
brake_max	Maximum brake command	(lbf)
lataccel_max	Maximum allowed lateral acceleration	(ft/s/s)
decel_tg	Target deceleration	(ft/s/s)
speed_tg	Target speed	(mph)
speed_min	Minimum speed target	(mph)
speed_max	Maximum speed target	(mph)
lanechange_time	The estimated time for a commanded lane change	(sec)

Table 2. Keyboard inputs

Key	Function
w	Drive at the speed limit
s	Brake to a stop
j	Adjust speed down by 5 mph
k	Adjust speed up by 5 mph
a	Change lanes to the left if possible
d	Change lanes to the right if possible
W	Set the next turn direction to straight
A	Set next turn direction to left
D	Set next turn direction to right
p	Pullover to the side of the road
1	Style coefficient = 0.0
2	Style coefficient = 0.2
3	Style coefficient = 0.4
4	Style coefficient = 0.6
5	Style coefficient = 0.8
6	Style coefficient = 1.0

Steering control is achieved by computing the lane offset error at the center of the car and at a look-ahead location calculated from a look-ahead headway time. The look-ahead distance is calculated from the vehicle's speed and the headway time. This distance is measured along two trajectories: the projected line along the vehicle's heading and the path along the lane. The latter may be curved if on a curved lane or corridor. Additionally, four points are sampled along the lane trajectory and the path radii are computed. The minimum of these radii is used to moderate the velocity given the constraints from the minimum lateral acceleration parameter.

Several miniSim RTEK cells have been defined to transmit automated driver signals during the simulation. These cells are documented in Table 3.

Table 3. Auto-driver cells

Parameter	Description	Notes
steer_kp	Proportional gain applied to lane offset error	overridden by gain scheduling
steer_kd	Derivative gain applied to look-ahead offset error	overridden by gain scheduling
steer_ki	Integral gain applied to integrated offset error	
steer_min	Minimum steering angle	(deg)
steer_max	Maximum steering angle	(deg)
lookahead_time	Time headway to use for look-ahead offset point	(sec)
lookaheadrate_lim	Rate limit for changing the look-ahead time	
throttle_kp	Proportional gain applied to velocity error	
throttle_kd	Derivative gain applied to acceleration	
throttle_ki	Integral gain applied to integrated velocity error	
throttle_fwd	Forward throttle signal gain	
throttle_min	Minimum throttle command	(0-1)
throttle_max	Maximum throttle command	(0-1)
brake_kp	Proportional gain applied to velocity error	
brake_kd	Derivative gain applied to acceleration	
brake_ki	Integral error applied to integrated velocity error	
brake_fwd	Forward brake signal gain	
brake_min	Minimum brake command	(lbf)
brake_max	Maximum brake command	(lbf)
lataccel_max	Maximum allowed lateral acceleration	(ft/s/s)
decel_tg	Target deceleration	(ft/s/s)
speed_tg	Target speed	(mph)
speed_min	Minimum speed target	(mph)
speed_max	Maximum speed target	(mph)
lanechange_time	The estimated time for a commanded lane change	(sec)

System Integration with miniSim™

Sometimes you need the miniSim to talk to other 'things'! This could be eyetrackers, touchscreens, buttons, gages, indicator lights, and things like that. This can be accomplished one of four ways:

Route Table

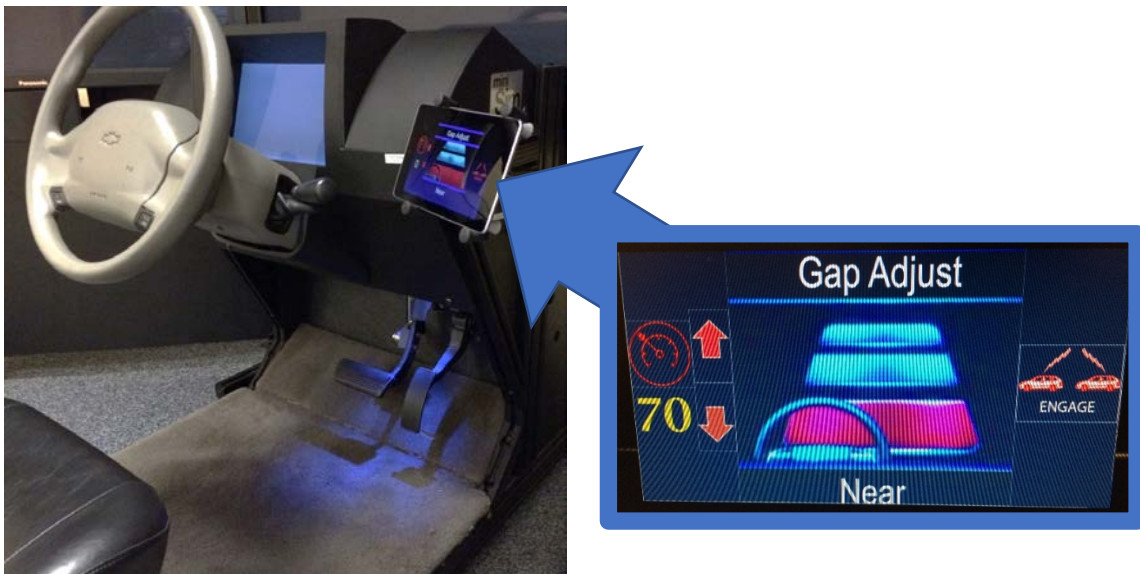
The miniSim can communicate with external devices, either NADS-supplied or user-developed. The primary interface is via a UDP datagram, transmitted over WiFi or LAN. These messages are set via a miniSim utility called the Route Table once per simulation frame. The Route Table defines:

- The miniSim cell variable name(s) to transmit and receive
- The destination IP address
- Scaling and offsets if desired

The function can be used to do the following:

- Trigger events in simulation
- Control miniSim AutoDriver
- Log data in miniSim
- Control external devices

An example is the use of a touchscreen to control the state of an ACC system:



Analog and Digital I/O

The hardware subsystem can be used to either transmit or receive data via Analog (0-5Vdc) or Digital (active low) inputs and outputs. Examples include:

- Operator Event Triggering. A user wanted a way for the operator to trigger an event by pressing button. The signal chain used is as follows:

Button ->USB I/O -> Hardware Subsystem ->Cell Variable ->Expression Trigger -> Action

The actions are those available in ISAT: visual or audible alert, lead vehicle braking etc.

- Interface with [BioPac™](#). Here a user wants to record physiological data with the BioPac and synchronize it with the simulator data. The solution was to use the hardware subsystem to map a miniSim variable (such as accelerator pedal position) to an analog output, which the BioPac can record.
- Interface via digital output. Many devices can utilize a digital input for data synchronization; typical ones are EEG systems, BioPac, and Eye Trackers. Here, the solution was to use the hardware subsystem to map a scenario trigger to a digital output, which the device can record. The typical signal chain would be:

Time Trigger -> Action: Write to Cell -> Hardware Subsystem -> Digital Output

User-Defined Simulator Subsystem

The miniSim subsystems are state machines, whose current states are controlled through the user interface. A user can create a custom subsystem to run with the rest of the MiniSim simulator. The ability to link user developed functions to the subsystem has many applications, including the incorporation of active vehicle safety systems or automation functions into the miniSim driver in the loop simulation environment. It is also possible to build custom MATLAB functions into static library files, include their accompanying C++ header files in the source code for the custom subsystem, and link the static libraries into the project the same way the libraries for the subsystem base class are linked in.

Custom subsystems can be added to the miniSim with little change to the rest parts of the system. The custom subsystem can be built from the subsystem base class which provides an inter-subsystem data communication interface and the functionalities of the real-time scheduling core that monitors state changes. The user layer registers input and output data elements, and executes custom functions that based on the current state of the simulator, interpret the input data and generate outputs.

NADS will provide code for a functioning example subsystem, including the required subsystem header and library files, as sample source code for a simple subsystem which does data input and output via the element registration scheme and handles state changes. The MiniSim software was written in C++ and built with Microsoft Visual Studio.Net solution and project files for the sample subsystem are included.

Custom Interfaces

Sometimes the best approach is to write a program that is launched by the miniSim when it is started. This program listens for miniSim's UDP datagrams (see Route Table) via the PC's internal loopback IP address, and communicates with an external device via another method such as an API. Examples include:

- Haptic Seat. Here we developed an interface to [Engineering Acoustics Inc.'s](#) (AEI's) Tactor Controller using their API to control up to 16 individual factors that we embedded in the driver's seat. The user creates cues (basically sequences of spatial and temporal vibrations) using EAI's [Tactor Creator software](#). These are downloaded to the controller and in use, the miniSim triggers the appropriate cue via scenario control using a Write to Cell action.



Haptic Seat Created using EAI's Tactor Control and API

- [Ergoneers and D-LAB](#). Here we worked with Ergoneers to develop an application that reads the miniSim's UDP data stream and converts it into a TCP/IP format for their D-LAB application to read. This allows the user to record simulator data (lane position, speed, steering angle, etc) as a network data stream along with eye tracking, video and other data fully synchronized for analysis in D-LAB.
- This approach has also been used to interface the miniSim with Eye Tracking systems using an API.

Virtual World API

This Logical Database API is to provide a simplified programmer interface into CVED, to serve as a method to extract road information at run time on the MiniSim. This API will not directly serve as an interface into the MiniSim, it will however be able to open the BLI file and run parallel with the MiniSim. As part of the delivery an example program that does provide a communication interface to the MiniSim, and utilizes this API will be delivered. The below figure shows the data-flow, where the example program receives the current state information including the position of the driver, the example program then uses this information to query the API.

Definitions:

CVED:	Correlated Virtual Environment Database is an API used internally at NADS to read the bli, and query the road network.
BLI:	Binary Logical road Interface, this is a file that contains all of the logical road information.
SOL:	The sol file provides static information about static or dynamic objects, such as the bounding box size
Corridor:	A corridor is a defined path through an intersection
Static Object:	Static objects include signs, and non-moving objects that are either part of the BLI, or inserted into ISAT by the scenario author.
Dynamic Object:	Dynamic objects are those objects that when created by running the scenario, have a physical presence in the simulation, and are updated by scenario control. These are either ADOs or DDOs.

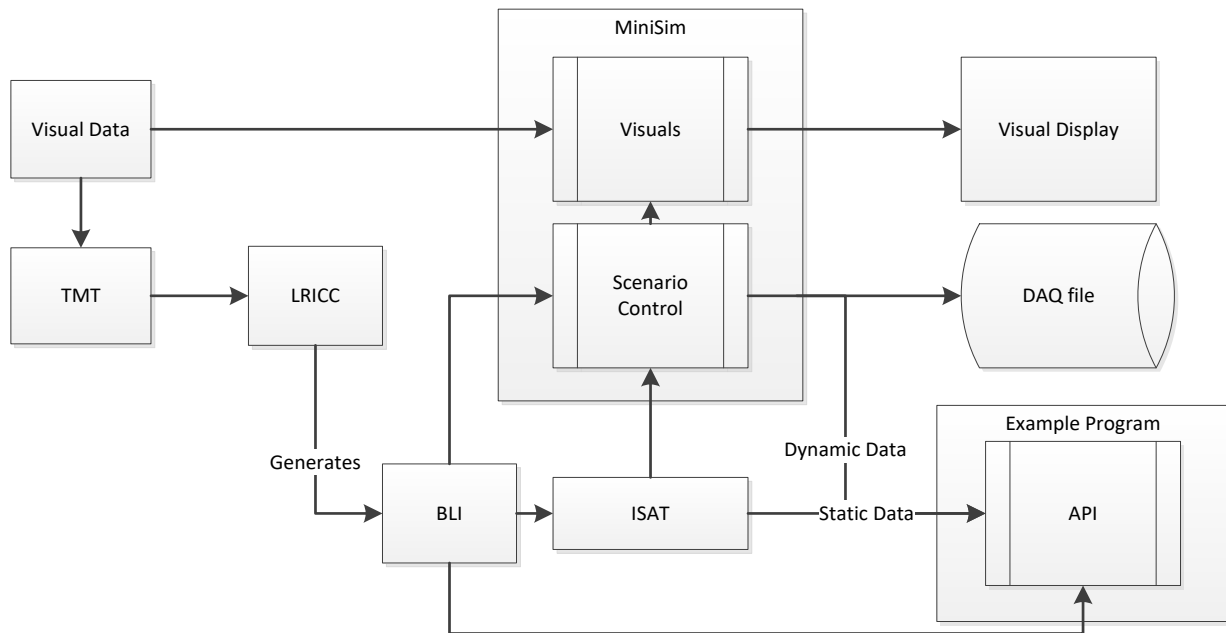


Figure 1 MiniSim Data Flow

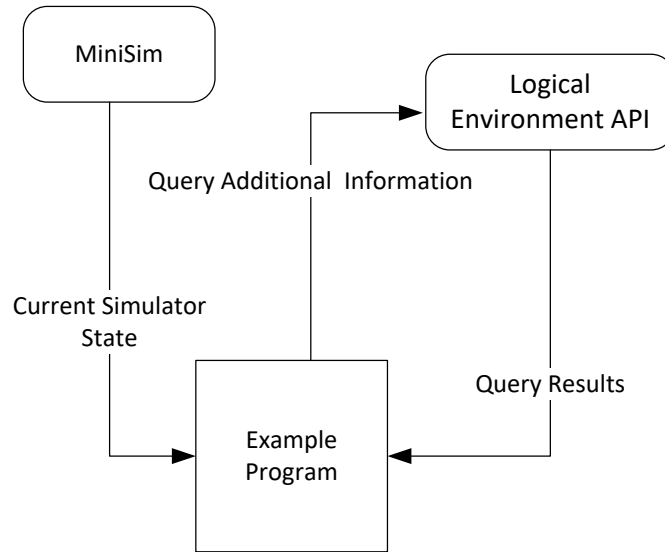


Figure 2 Data flow for API

The API is developed in C++; it will be delivered as a library file built in VS 2010, with a header files. The API is object oriented in design and will use Hungarian notation (classes begin with C, member variable will begin with m_). The example program will be provided as a source code distribution, with a project and solution file for VS 2010. This program will be developed with C++. The operation of the example program will require network connectivity with the MiniSim. The API will require in the installation directory of the executable:

- a copy of the BLI
- a copy of the SOL file
- a copy of the scenario file

Description of API Functionality:

The API provides an interface that will allow a user to query logical information about the driving environment, such as lane width, distance to next intersection. The API will accept x,y,z locations and provide road related data about said location. This shall have real-time level performance. It also shall be possible to use this API in a system that is not directly connected to the MiniSim.

The example program will communicate with the MiniSim, and will utilize at least one version of every API available, duplicate calls that only vary in call type (i.e. float vs. double) may only have 1 version called. This program will provided as an example of the API, and will provide validation of the API.

Road Information

The API will provide a series of functions that allow the user to input an X,Y,Z location and get logical information at that location, including lane width, road marking type , surface type . Only a minority of tiles have tags (or attributes) that allow road marking information to be queried by CVED, hence this API will only provide road marking information from BLIs that a marked with tags for road markings. Should a road position not have valid markings tag, a return code will indicate no information was available.

GetLaneWidth()

This will return the current width of the current lane at a given XYZ location.

GetRoadMarkings()

This will return lane marking type, on the right and left of the lane. This shall also provide distance from center line to inside of lane markings at a given XYZ location.

GetLaneLayout()

This function shall return the number of lanes, the direction of each lane, and meta-data about the lane, given a XYZ position; this will not function in intersection.

GetLaneSurfaceType()

This function will return the road surface type at the current location. Documentation will be provided as to what the surface type codes indicate.

GetRoadTrace()

This function will take the current position, and a distance, and return an array of x,y,z points and lane width, following road. A version of this function will use the driver's path as specified by the scenario, and use that to navigate any intersections that the function may encounter. Another form of the function will take a direction in, and will navigate the intersection using a given direction (right, left, straight). Another version will take a corridor id, and provide a trace through the corridor.

GetOncomingIntersection()

Forms of this function will return: the distance to the next intersection 0, if currently located on an intersection, and list of connected corridors, and their respective turn directions.

GetOncomingHaltLine()

This function will return the distance and x,y,z, location of the next oncoming halt line, version of this function will take the drivers path to determine what halt line to stop at.

GetOncomingMergePoints()

This function will take a distance, and position, it will project forward, and return distances to, and x,y,z location for points where two corridors begin to overlap.

GetLightState()

This function will take in location information and the traffic light state data past from the MiniSim, and will indicate what the traffic light state is for said location.

GetCurvature()

This function will take xyz location, and return the radius of curvature for said location.

GetGrade()

Given an x,y,z get the road grade of the road in degrees.

ChaneLanes()

Given a position (XYZ) on a road segment (not in an intersection) and a lane offset(+ meaning lanes to the right, negative numbers mean to the left, this function will return center of the lane indicated by the lane offset or an error code indicating why it cannot changes lanes(i.e. the lane does not exist, or is an oncoming lane).

Static Object Functions

GetStaticObjects()

Given a bounding box, this function will return a list of static objects, this will include x,y,z information and sol ID. Most signs will provide a x,y,z location, where this position will provide the bottom of the post of the sign (for signs with a single post). The height of the sign will be derivable from the GetSolInfo. However not every sign's position is directly related to its position in the BLI (some signs such as interstate with many components do not match the actual position as to ease authoring and viewing the sign in ISAT). Some guidance will be provided as what signs can have their positions directly derived from the bli.

Dynamic Object Functions

UpdateDynamicObjects()

This function takes in the dynamic object data outputted from the MiniSim, and adds entries into CVED for each object.

GetDynamicObjects()

Given a bounding box, this function will return a list of dynamic objects, this will include x,y,z information and sol ID.

GetTerrainObjects()

Given a bounding box, this function will return a list of terrain objects; terrain objects provide information about the location of additional terrain information, such as parking lots, or buildings. This tends to only be a much smaller subset of what is in the visual database, this API will only allow access to terrain objects that have been placed in the BLI file.

GetSolInfo ()

Given sol ID (that which is part of the dynamic object data), this function will return information from the SOL file, including the minimum bounding box for the object.

Example program

An example program is provided, this program:

Utilizes every query available to the API

Every query function accessible to the API shall be called at least one

MiniSim Connectivity

The sample program shall receive all data that the query API may utilize, and properly load the data into the API's objects.

Full Source code and project file will be provided

The source code and project files will be provided for the sample program, the end user shall be able to build this program using Visual Studio 2010.

Linking NADSdyna with Matlab Simulink

In some advanced applications, it is possible for a user to create their own low-level behaviors similar to the AutoDriver functions in Simulink and link these to the miniSim's vehicle dynamics subsystem (NADSdyna). Please contact NADS for more details.

Virtual Environments

The two main components of the miniSim™ virtual environments are the roadmap and model libraries. The combination of these two libraries results in the most realistic driving environment. An example of a scene is presented in Figure 21. The following sub-sections describe the contents of the roadmap and model libraries in more detail.



Figure 1 - A high resolution model and driving environment from the miniSim™ driving simulator

NADS also has proven capabilities in the development of geo-specific virtual environments. These are virtual environments that recreate actual locations in the world to a very high degree of accuracy (within an inch). An example of a geo-specific virtual environment is shown in the following figure. This is a virtual version of an actual dirt racetrack from West Liberty, Iowa.



Figure 2 - The West Liberty raceway (left is the actual location, right is the virtual environment)

NADS miniSim™ Included Tiles and Databases (*Right-Hand-Traffic Only*)

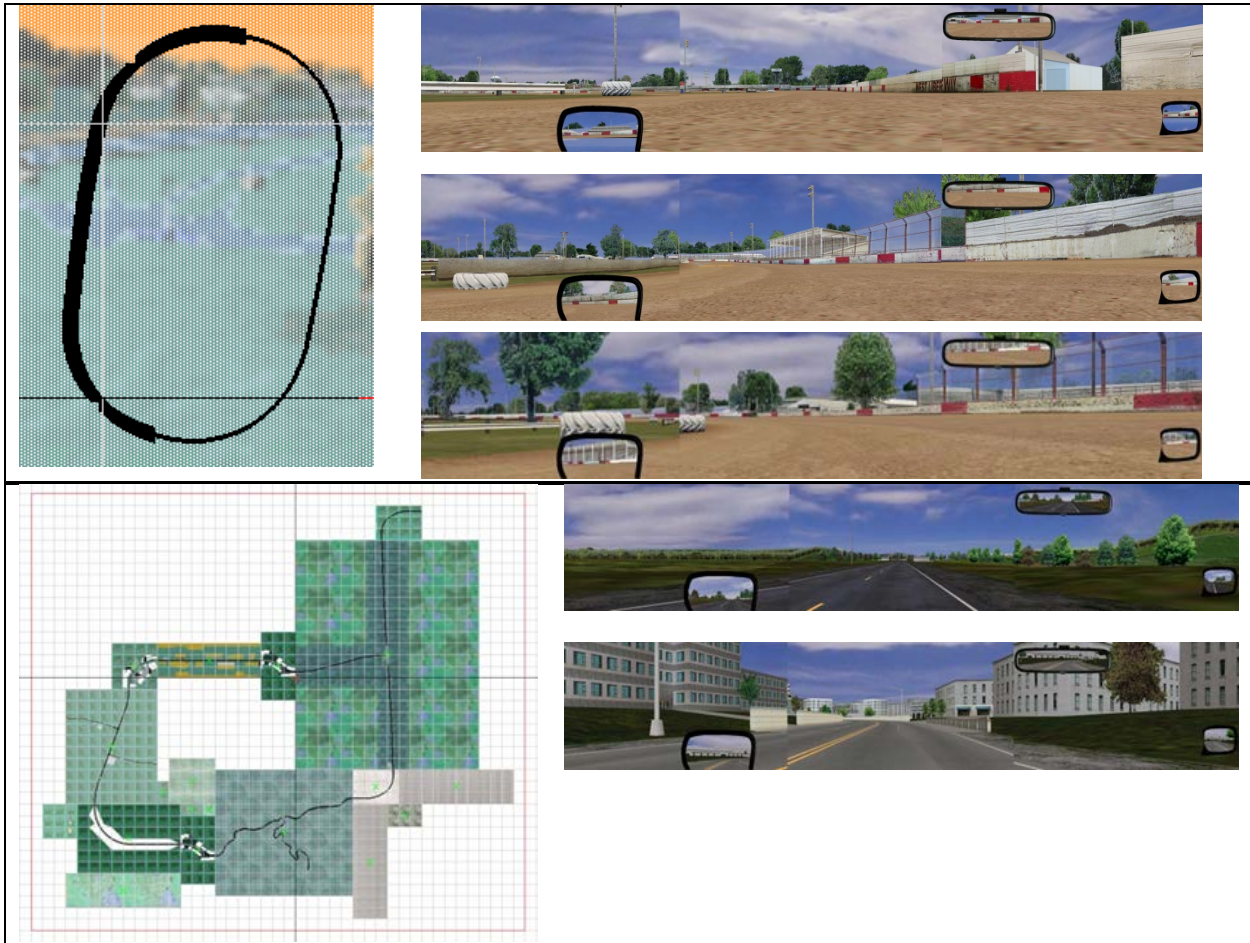
There are approximately **190** database tiles that are included with the miniSim. These represent road segments that can be assembled into custom databases of your own design using the NADS Tile Mosaic Tool (TMT).

Also included are the following databases that have already been assembled and tested:

NADS TMT v1.2 installed databases

- **demo** : Databases developed for demonstration purposes
 - **nadsdemo_dsc**: DSC conference demo. Contains urban, rural, and mountain/snow.
 - **nadsdemo_geospecific**: Geo-specific capabilities demo (West Liberty Racetrack)
 - **nadsdemo_kiosk**: Outreach database; 3 separated databases on one terrain, cannot drive from one to the other
- **ESC_Dry_05**: Rural 2 lane database developed to test electronic stability control; open-loop database with decreasing radius curve at north end of terrain
- **ESC_HvyT**: Interstate and Rural 2-lane database developed to test electronic stability control.
- **multipleEnv_01**: Combination city, suburb, freeway, rural database developed to design and evaluate levels of impairment
 - **day_night** : Original study was night only, this version is not the same as the study version due to changing the night version tiles with day tiles.
 - **snow** : This version is not the same as the study version, the rural tiles are snow covered, otherwise the same as the day_night version
- **rural_long** : Rural 2 lane, open-loop database developed to test antihistamine response, repurposed in other studies as well.

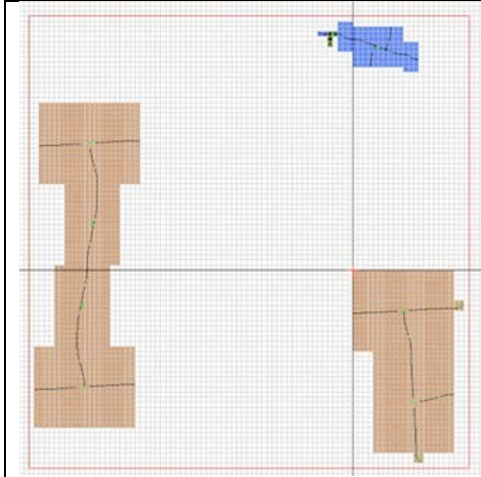




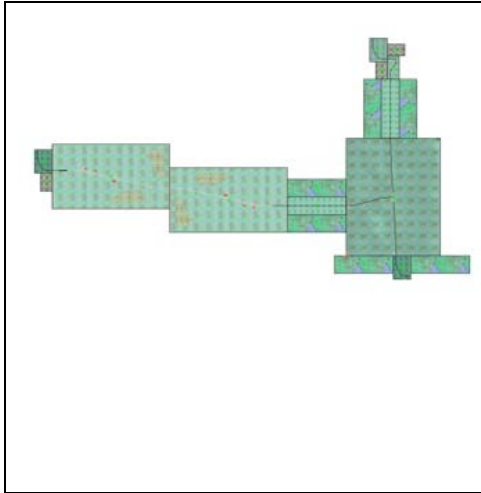
Nadsdemo_dsc database

Nadsdemo_geospecific

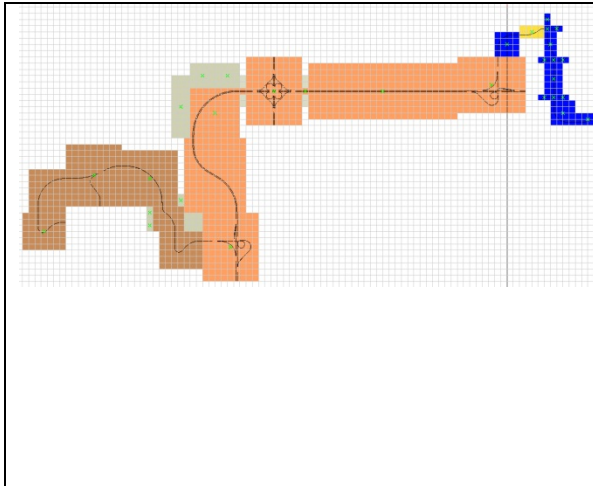




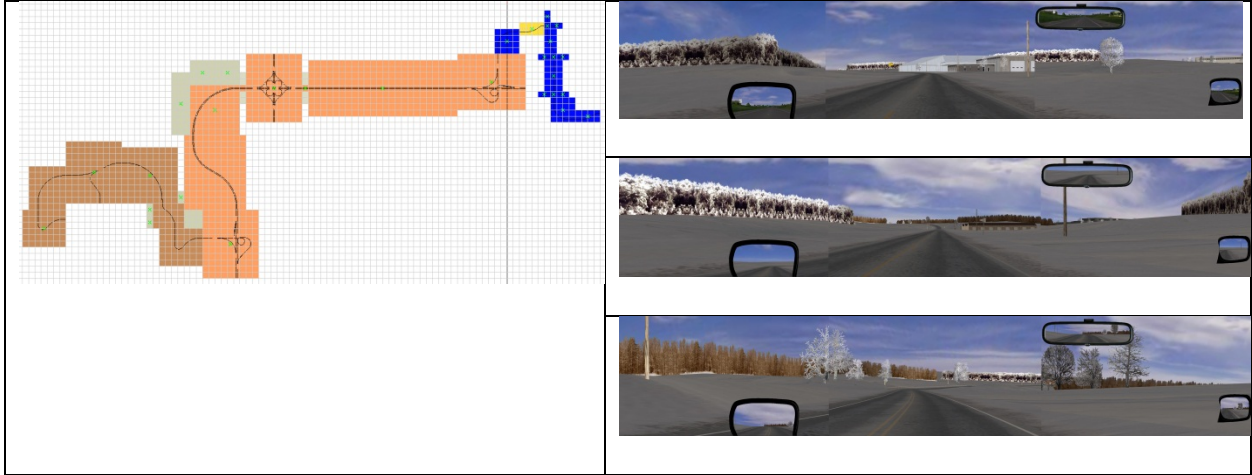
Nadsdemo_kiosk



ESC_Dry_05



MultipleEnv01_Day_night



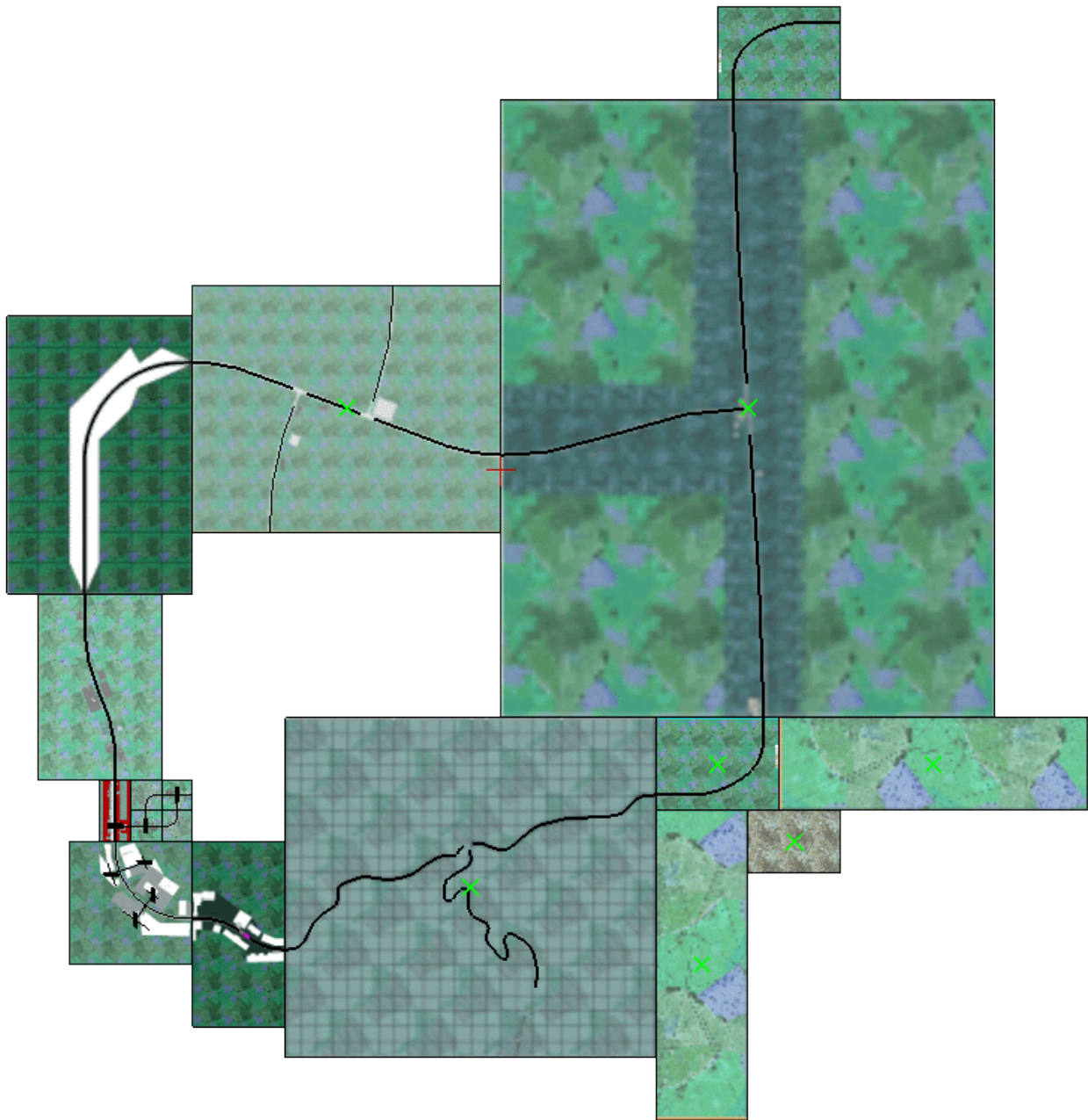
MultipleEnv01_Day_snow



Rural_long

NADS Demo Database Details

A short roadmap that combines a number of different types of driving environments. Urban, rural scenes are followed a mountainous scene with switchbacks. Day and night-time driving are also featured.





Terms and Conditions

The *miniSim SUPPLEMENTAL AGREEMENT TERMS* follow. **These terms supersede the terms of any Purchase Order submitted to the University.** If a purchase order is used, please include the following statement: ***“The MiniSim Supplemental Agreement Terms supersede the standard PO terms and conditions.”***

The above fixed-price quote is fixed price and is valid for 60 days from the proposal date.

The University of Iowa
Division of Sponsored Programs
2 Gilmore Hall
Iowa City, Iowa 52242
Phone: 319-335-2123
Fax: 319-335-2130
E-mail: dsp-contracts@uiowa.edu

DUNS: 062761671

Please cc: andrew-veit@uiowa.edu

Contract Period: The Contract Period is 12 months from the date of signature and may be extended by written notification.

Payments: NADS will invoice recipient for the support and maintenance fee upon contract execution, and annually thereafter.

Contact: National Advanced Driving Simulator
Joe Meidlinger
joseph-meidlinger@uiowa.edu
(319) 335-4302
2401 Oakdale Blvd
Iowa City, IA 52242

UI Grant Accounting will invoice the remainder of the contract amount upon delivery of simulator.

Contact: Grant Accounting Office
118 S. Clinton Street
Iowa City, IA 52242
Phone: 319-335-3801

MiniSim SUPPLEMENTAL AGREEMENT TERMS

These Supplemental Agreement Terms supplement the terms of the proposal to which they are attached and are effective on the date of signature between the parties named in the proposal, a "Sponsor" and The University of Iowa, Iowa City, Iowa, a non-profit educational institution ("University").

ARTICLE 1 – Definitions

The parties agree to the following:

- 1.1 "Project" is as described in the attached proposal.
- 1.2 The contract period is as described in the attached proposal and may be extended by written agreement of the parties.
- 1.3 "NADS Software" is individually and collectively the software provided by University to Sponsor for Sponsor's use under the Project.

ARTICLE 2 – Contract Performance

University shall initiate performance of the Project promptly after the effective date of this Agreement, and shall use reasonable efforts, care, and diligence to perform such Project in accordance with the terms and conditions of this Agreement.

ARTICLE 3 – Grant of Rights

- 3.1 NADS Software is the property of University and is to be used by Sponsor solely for research purposes (human factors research) only at Sponsor's institution, only under direction of the Sponsor's Scientist(s), and as long as the yearly maintenance agreement is renewed.
- 3.2 The Sponsor's Scientist(s) agrees not to transfer or provide the NADS Software to anyone else at Sponsor's institution or any third party without prior written consent of University. The NADS Software being used is further restricted to a single computer.
- 3.3 The Tile Mosaic Tool (TMT) software is made available under this Agreement under permission from Rockwell Collins Simulation and Training Systems (RC/STS). In addition to the other terms and conditions of this Agreement, the terms and conditions of Exhibit A apply to Sponsor's possession and use of TMT. Should any of the terms and conditions of this Agreement conflict with the terms and conditions of Exhibit A, the terms and conditions of Exhibit A shall govern.

ARTICLE 4 – Costs, Billings, and Other Support

- 4.1 Total costs to Sponsor shall not exceed the amount in the proposal without written approval by the Sponsor. The hours reflected in any proposal or resultant contract are provided for estimation purposes only. In accordance with the principles defined in the Uniform Guidance, 2 CFR, The University of Iowa maintains documentation of effort worked for its professional

and scientific staff on a percentage basis. In the event hourly employees are utilized for the project, hourly records are maintained for those employees.

- 4.2 Payment terms are in U.S. Dollars. The balance is due on receipt of an invoice to the address indicated on the invoice.
- 4.3 In the event of early termination of this Agreement by Sponsor, Sponsor shall pay all costs accrued by University as of the date of termination, including non-cancelable obligations and contracts incurred prior to the date of termination.

ARTICLE 5 – Publicity

- 5.1 Neither party shall use the name of the other, nor of any member of their staff, in any publicity, advertising, or news release without the prior written approval of an authorized representative of the other party. Notwithstanding the foregoing, University may disclose the name of Sponsor, and the existence and general nature of the agreement to meet its reporting requirements.
- 5.2 Sponsor agrees that the National Advanced Driving Simulator at the University receive appropriate recognition consistent with academic standards in any publication or presentation resulting from the use of the MiniSim.

ARTICLE 6 – Confidentiality

- 6.1 It is the responsibility of Sponsor to mark or otherwise identify in writing prior to submission any information considered confidential that it deems necessary to share with University.
- 6.2 Oral disclosures of confidential information shall be identified as confidential at the time of disclosure and confirmed in writing within ten (10) business days of the disclosure.
- 6.3 University shall withhold it from publication, and in all other respects it shall be maintained by University as confidential and proprietary to Sponsor for a period of ten (10) years from disclosure. University shall have no such obligation with respect to any portion of such information that:
 - a) Is, or later becomes, generally available to the public by use, publication or the like, through no fault of University;
 - b) Is obtained on a non-confidential basis from a third party who disclosed the same to University;

- c) University already possesses, as evidenced by its written records, predating receipt thereof from Sponsor; or
- d) Is required by law to be disclosed.

6.4 All documentation concerning NADS Software submitted to Sponsor in accordance with this agreement shall be treated by Sponsor as confidential.

ARTICLE 7 – Insurance

7.1 Each party shall be liable for any and all claims for wrongful death, personal injury or property damage attributable to the negligent acts or omissions of that party and the officers, employees, and agents thereof.

7.2 University shall be responsible and agrees to pay for any and all claims for wrongful death, personal injury or property damage directly resulting from the negligence of University, its officers, employees and agents, and arising from activities under this Agreement to the full extent permitted by Chapter 669, Code of Iowa (2011), which is the exclusive remedy for processing tort claims against the state of Iowa.

ARTICLE 8 – Indemnification

Neither party shall indemnify the other party.

ARTICLE 9 – Governing Law

This Agreement shall be governed by the laws of the State of Iowa, excluding its conflict of laws provisions.

ARTICLE 10 – Agreement Modification

Any agreement to change the terms of this Agreement in any way shall be valid only if the change is made in writing and approved by mutual agreement of authorized representatives of the parties.

ARTICLE 11 – Warranties and Liability

11.1 University in no way guarantees results of work performed pursuant to this Agreement, AND MAKES NO WARRANTIES, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, AS PART OF THIS AGREEMENT.

11.2 Force Majeure. No failure or omission by a party hereto in the performance of any of its obligations under this Agreement shall be deemed a breach of this Agreement, nor shall it create any liability, if the same is solely due to a cause or causes beyond the reasonable control of the performing party; provided that the party so affected shall use reasonable efforts to avoid or remove such causes of nonperformance and shall continue performance hereunder with the utmost dispatch whenever such causes are removed.

Exhibit A









The Tile Mosaic Tool (TMT) tool is supplied and supported by University under permission from Rockwell Collins Simulation and Training Systems (RC/STS) and is made available for Sponsor's use under the following terms and conditions:

- Recipients of TMT may only use it internally for non-commercial educational or research purposes.
- Recipients may not distribute TMT to other third parties.
- RC/STS retains title to TMT.
- Recipients shall not de-compile, reverse engineer or develop any derivative works from the TMT.
- TILE MOSAIC TOOL IS PROVIDED IN AN "AS-IS" CONDITION. IN NO EVENT SHALL RC/STS BE LIABLE TO RECIPIENT FOR ANY CLAIMS FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL LOSSES, DAMAGES, OR EXPENSES ARISING OUT OF THIS AGREEMENT, OR THE USE OR PERFORMANCE OF THE TILE MOSAIC TOOL, REGARDLESS OF THE FORM OF ACTION, WHETHER IN CONTRACT, TORT, OR UNDER ANY STATUTE, INCLUDING NEGLIGENCE OR OTHERWISE, EXCEPT FOR BODILY INJURY CAUSED BY NEGLIGENT ACTS OR WILLFUL OMISSIONS OF RC/STS.









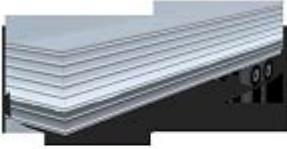








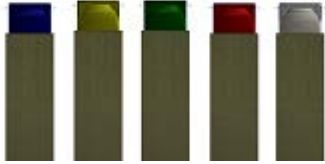
Scenario Object Library (SOL)

The NADS Scenario Object Library (SOL) features a comprehensive library of vehicle models, signs and obstacles. Each vehicle has the capability to exhibit intelligent traffic behavior. Most signs feature a number of options to allow the user to customize the training environment. In total, the model library features over 300 unique elements. The model library is included with the miniSim™.

Vehicles

 <p>Ambulance</p>	 <p>Audi A8</p>	 <p>Chevy Blazer</p>
 <p>BMW</p>	 <p>Cadillac Escalade</p>	 <p>Cement Truck</p>
 <p>Beverage Truck</p>	 <p>John Deere Combine</p>	 <p>Cadillac Deville</p>

  <p data-bbox="321 548 480 583">Dump Truck</p>	  <p data-bbox="740 548 906 583">Ford Escape</p>	  <p data-bbox="1097 548 1393 583">Freightliner Semi Truck</p>
  <p data-bbox="334 1016 469 1052">Ford F150</p>	  <p data-bbox="721 1016 922 1052">Ford Expedition</p>	  <p data-bbox="1149 1016 1338 1052">Garbage Truck</p>
  <p data-bbox="305 1486 496 1522">Landrover LR2</p>	  <p data-bbox="732 1486 915 1522">US Mail Truck</p>	  <p data-bbox="1162 1486 1325 1522">Dodge Neon</p>

  <p>Peugeot</p>	  <p>Phone Truck</p>	  <p>Police Car</p>
  <p>Saturn Vue</p>	  <p>Semi Trailer</p>	  <p>Ford Taurus</p>
  <p>Lincoln Town Car</p>	  <p>John Deere Tractor</p>	  <p>Cargo Truck</p>



Control Signs



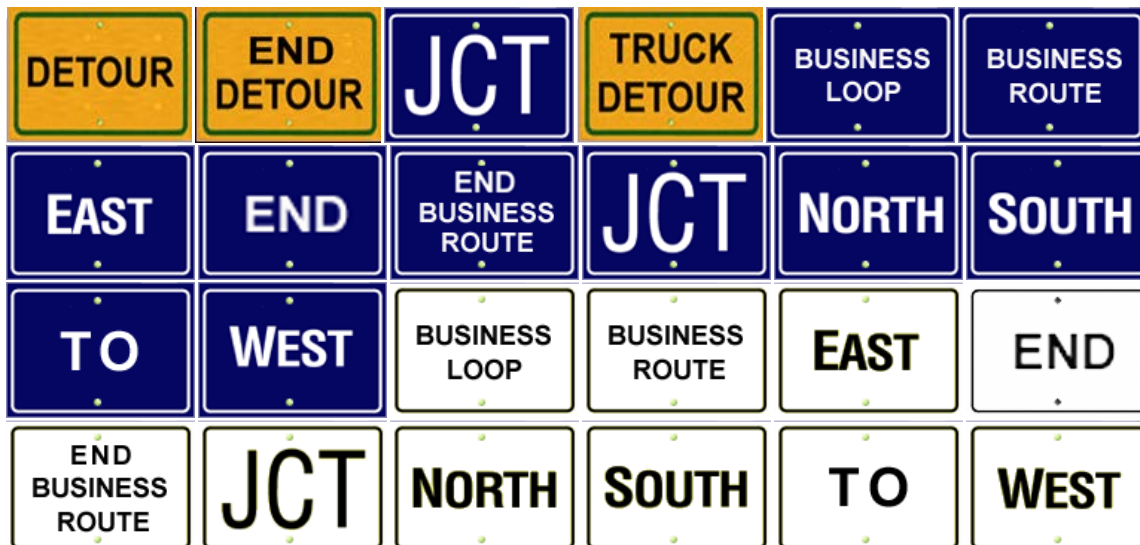
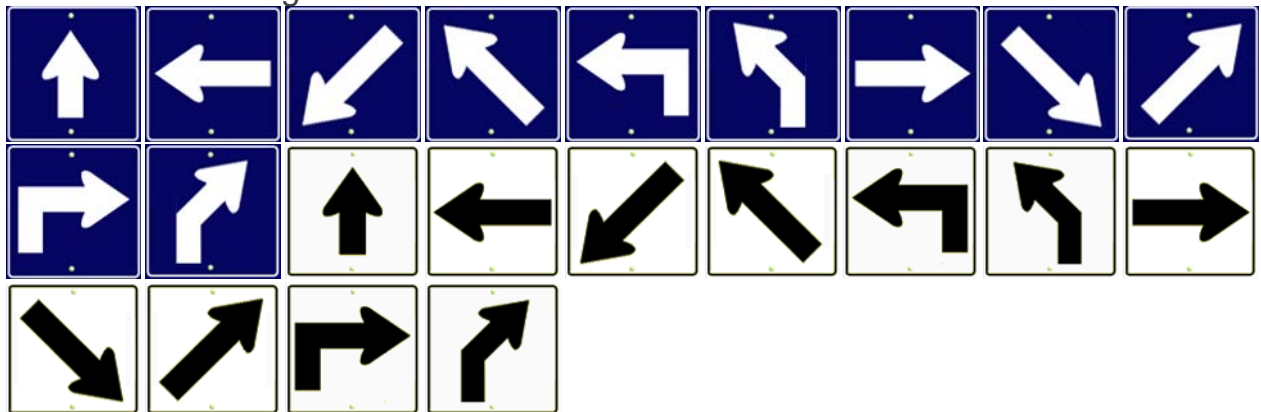
Warning Signs



Construction Signs

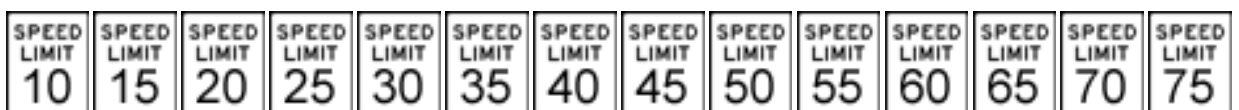


Route Direction Signs



Speed Limit Signs

Speed limits signs for both positing speed limits on roads and on ramps.




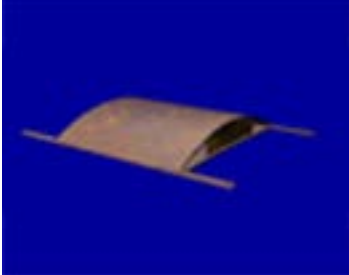
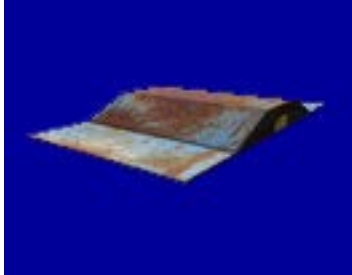
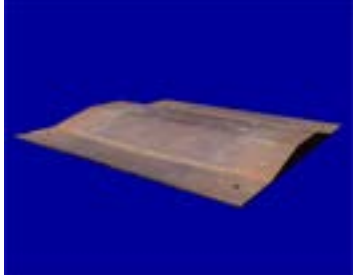

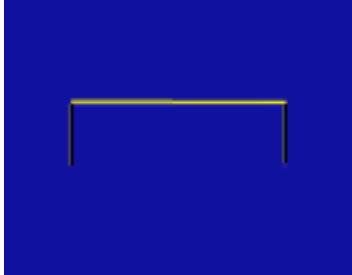

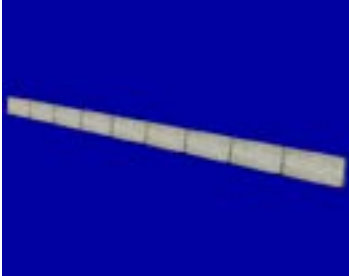


















Interstate and Highway Signs


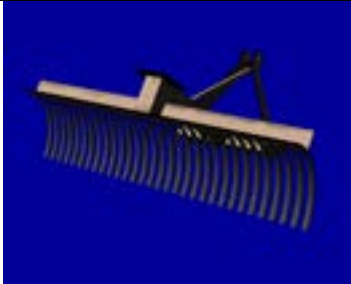
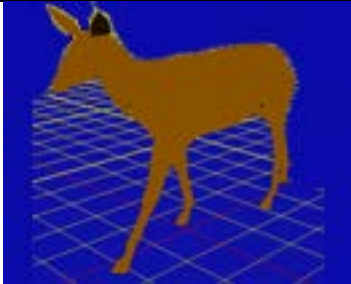












Objects

 <p>Barricade</p>	 <p>Bump Hump</p>	 <p>Bump with Large Plateau</p>
 <p>Bump with Small Plateau</p>	 <p>Pole 1</p>	 <p>Pole 2</p>
 <p>Concrete Barrier (24')</p>	 <p>Concrete Barrier (100')</p>	 <p>Cone</p>

		
<p>Crash Barrel</p>	<p>Diagonal Stripes</p>	<p>Drum</p>
		
<p>Halt Line with White/Red</p>	<p>Bouncing Ball</p>	<p>Desk (falling off Vehicles)</p>
		
<p>Foam Pad</p>	<p>Suitcase</p>	<p>Reflector Pole</p>
		
<p>Parking Meter</p>	<p>Pedestrian Clutter</p>	<p>Pedestrian Clutter 2</p>
		

<p style="text-align: center;">Smoke</p>  <p style="text-align: center;">Dethatch</p>	<p style="text-align: center;">Street Light</p>  <p style="text-align: center;">Stone rake</p>	<p style="text-align: center;">Traffic Light</p>  <p style="text-align: center;">Traffic Channelizer</p>
 <p style="text-align: center;">Deer</p>	 <p style="text-align: center;">Dog</p>	 <p style="text-align: center;">Walker 1</p>
 <p style="text-align: center;">Walker 2</p>	 <p style="text-align: center;">Walker 3</p>	 <p style="text-align: center;">Ped Group 1</p>
 <p style="text-align: center;">Ped Group 2</p>	 <p style="text-align: center;">Ped Group 3</p>	 <p style="text-align: center;">Walker 4</p>



Walker 5



Phone Truck



Road Construction 1



Road Construction 2



Road Construction 3



Road Construction 4



Road Construction 5



Taxi



Road Construction 6



Road Construction 7



Road Construction 8



Road Construction 9

		
Road Construction 10	Road Construction 11	Road Construction 12
		
Walker with Backpack	Dead Deer	Old Tire
		
Fendt Tractor	Tree	

City Signs

Ames	Cheboygan	Galesburg	Lees
Aurora	Chicago	Gotham	Liberty
Baldwin	Dallas	Hannah	Louisville
Baltimora	Denver	Homestead	Marion
Benton	Dunkirk	Innsbruck	Mottville
Bloomington	East Conklin	Jefferston	New Hope
Boston	Franklin	Kalamazoo	Omstead
Cadillac	Frontier	Kansas City	Otsego
Carbondale	Galena	Lansing	Owasso

Park City
Parrish
Portage
Priestly
Quigley
Rapid City

Ringgold
Riverside
Sagatauk
San Ruis
Shelby
Streeter

Taylor
Traverse City
Trayer
Underwood
Vancouver
Vinton

Warren
Westchester
Ypsilanti
Zurich

Street Signs

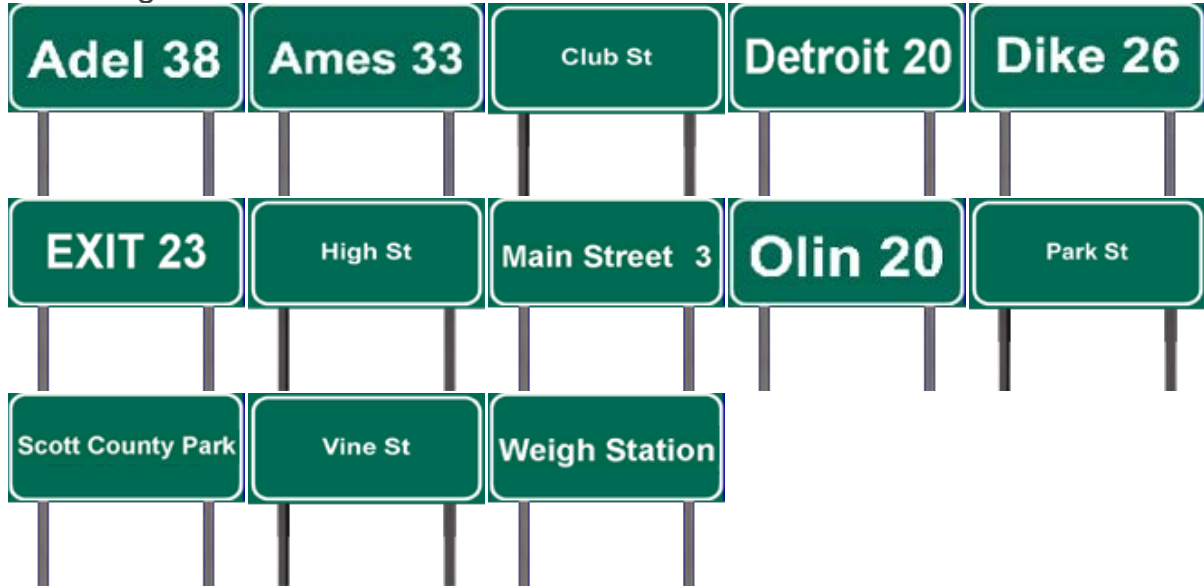
Adams Ave
Adams Dr
Adams Ln
Adams Rd
Adams St
Adams Way
Cobb Ave
Cobb Dr
Cobb Ln
Cobb Rd
Cobb St
Cobb Way
Fox Ave
Fox Dr
Fox Ln

Fox Rd
Fox St
Fox Way
Holt Ave
Holt Dr
Holt Ln
Holt Rd
Holt St
Holt Way
Martin Ave
Martin Dr
Martin Ln
Martin Rd
Martin St
Martin Way

Neff Ave
Neff Dr
Neff Ln
Neff Rd
Neff St
Neff Way
Park Ave
Park Dr
Park Ln
Park Rd
Park St
Park Way
Queen Ave
Queen Dr
Queen Ln

Queen Rd
Queen St
Queen Way
Taylor Ave
Taylor Dr
Taylor Ln
Taylor Rd
Taylor St
Taylor Way

Guide Signs



Rail Functionality

New functionality has been developed to support railroad crossings. This functionality includes new railroad vehicles, objects, tiles and operational capability. This implementation includes typical railroad crossing objects, terrain rail objects and associated signs.

Rail Vehicles

A limited set of rail vehicles has been implemented. These models are designed to operate along straight tracks only; the train model which consists of several cars does not articulate as the train traverses curves. The rail vehicles may be used as static, stationary objects as would be found on a railroad siding or as deterministic dependent objects (DDOs). DDOs require a travel path to be defined. This mechanism permits the rail vehicle to travel both directions along what is essentially a one-way road (track). Without the path, vehicles will always travel downstream, from the origin of the track towards the end of the track.

railveh_train_01



railveh_boxcar



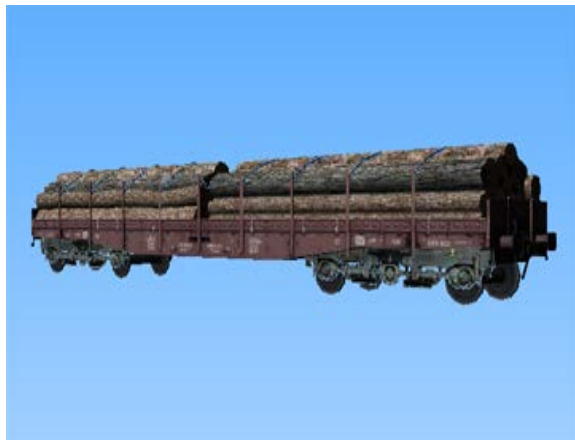
railveh_stock_car



railveh_coal_car



railveh_flatcar_logs



railveh_flatcar_steel_rolls



railveh_tanker_Gold_West

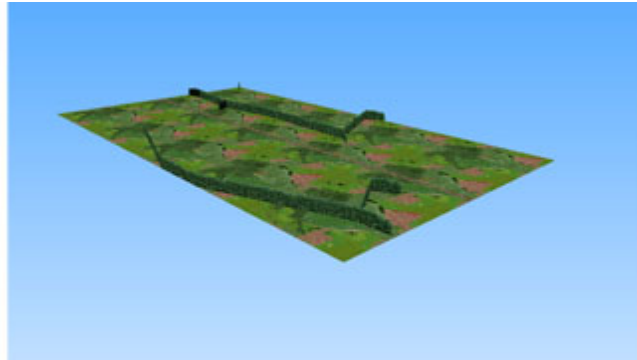


railveh_Union_Pacific

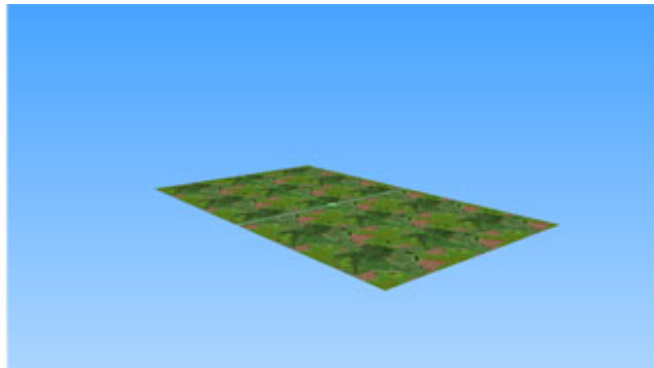


Railroad Tiles

The following railroad tiles are available: General-purpose rail tiles: 1X straight (single track), 3X straight (triple track), and at-grade crossings for rural and city environments. There are single and triple at-grade crossings, and one blank 1X and 3X rural crossing to permit total control over the crossing environment.



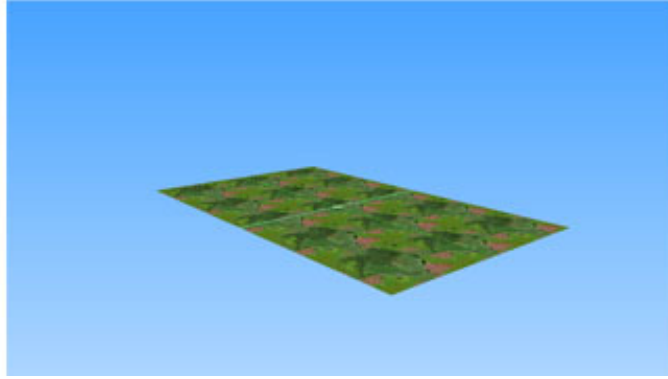
Railway_1x_straight



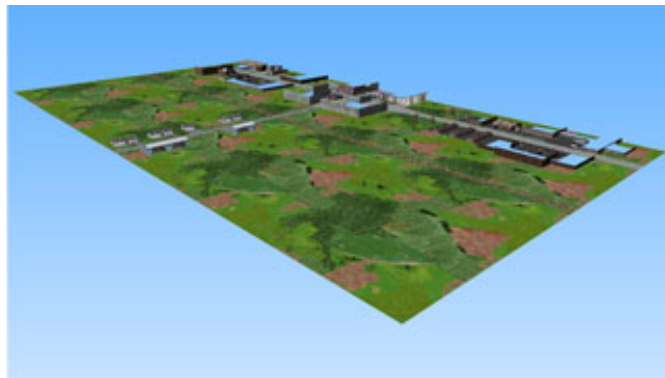
Railway_1xgrade_rural_01



Railway_1xgrade_rural_01 Overview



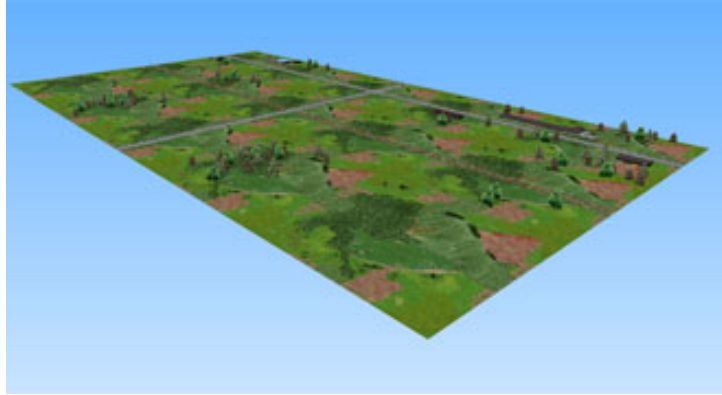
Railway_1xgrade_rural_02



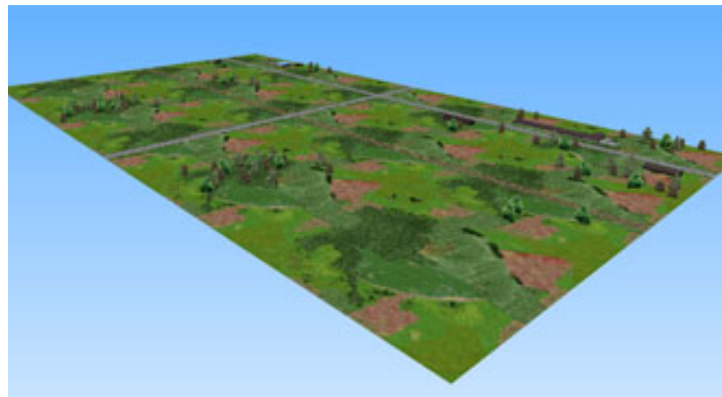
Railway_3xgrade_city_01



Railway_3xgrade_city_01 Overview



Railway_3xgrade_rural_01



Railway_3xgrade_rural_02