WaveTrain™
wave optics made easyier

Quick Reference

Version 2025A

Jason A. Tellez, Ph.D

MZA Associates Corporation
1360 Technology Ct, Ste 200, Dayton, OH 45430-2211
http://www.mza.com
(937) 684-4100, WT-Support@mza.com

The revision date is August 26, 2026

Contents

List of Figures

List of Tables

What's New

Compatibility Considerations

This release of WaveTrain™ has been tested on Windows with Visual Studio 2019 & 2022 and Matlab R2023b and will be compatible with that version and newer. However, WaveTrain™ 2025A may still work with older versions of Visual Studio and Matlab. On Linux, WaveTrain™ has been tested on Red Hat Enterprise Linux (RHEL) 9.0 and Ubuntu 22.04LTS and Matlab R2022b.

New Features

MZA is always looking for better and more useful ways to take advantage of the capabilities that WaveTrain™ offers. In that spirit, some new capabilities added to WaveTrain™ includes

- A new set of components for incoherent imaging based on the previous Light Tunneling techniques
- Improved third party library integration to include specifying locations using environment variables, simplifying the build process and improving portability
- Added an option to build without Matlab integration
- Updated drop down menu feature for selecting the build type in the Tempus Runset Editor (TRE)
- New build selection for creating a dynamic library version of the runset for interaction with the RunsetAPI set of functions
- Conductor application for interacting with RunsetAPI builds (BETA)
- (WTC) (BETA), a new way of building systems and running them, in some cases without the need for a C++ compiler or Matlab
- Updated examples

Bug Fixes

WaveTrain™ is continuously improving based on in-house use and customer feedback. As a result the following bug fixes have been addressed since the last release

- Fixed an issue where the gaussian random number generator may fail to work correctly
- Fixed an issue where conflicting libraries may interfere with WaveTrain™ execution from the TRE
- Fixed an issue where aogeom was not saving the subaperture locations correcly when the annulus is zero
- Fixed an issue where an unconnected polarizer component generates a false output wave
- Updated the memory allocation for the TSE to help prevent crashes when the TSE is open for a long time
- Fixed a bug where Tilt commands with NANs would cause the simulation to fail
- Added an AcsAtmSpec constructor that can be used to remove the "no altitudes defined" warning the the platform and target altitudes are available
- Fixed an issue where building a Mex function from the TRE would fail
- Fix a bug where thermal blooming phase was not being applied to non heating waves in some situations
- Fixed Matlab function tmxmkdll.m to use the new build process for creating MEX functions
- Fixed an issue where certain libraries may be left off the build link list for some users
- Fixed issue in Kerr propagation where extra propagation steps were being done in certain situations

1. Introduction

WaveTrain™ is a code suite that provides computer modeling of optical propagation through the turbulent atmosphere, and the modeling of associated optical imaging, beam control, and adaptive optics (AO) systems. WaveTrain™ provides a connect-the-blocks visual programming environment in which the user can assemble beam lines, control loops, and complete system models, include closed-loop AO systems. WaveTrain™ also provides graphical interfaces for setting up adaptive optics geometries and turbulence profiles through a handful of MATLAB® based utilities along with a spreadsheet-style interface for setting up and executing parameter studies. The generated output, comprising imagery, complex-field quantities, control signals, etc. can be inspected in a WaveTrain™ viewer or can be loaded into MATLAB® for further analysis and post processing.

This User Guide is designed to serve as a tutorial introduction to WaveTrain™ for new users, and a reference for more experienced users. The User Guide attempts to be as comprehensive as possible in documenting the scope, usage rules, and assumptions of the WaveTrain™ code. In doing so, we may occasionally digress into short tutorials on topics in optical propagation theory and AO but for detailed theory background we refer the reader to the 2.6. WaveTrain™ is a complex code, and its pieces have be constructed and assembled by numerous contributors at MZA Associates. Generation of documentation is an ongoing effort, consequently not all elements of WaveTrain™ are documented to the same level of details. When users have questions regarding WaveTrain™ we ask them to proceed as follows

- Search the WaveTrain™ User Guide
- Check the MZA Website for updates
- Check the individual module documentation in the WaveTrain™ Components and Effects Library
- Depending on level of support contact MZA by email or phone
- If a suspected bug is found, feel free to contact MZA regardless of level of support (or lack thereof)

WaveTrain™ is built on tempus, a general-purpose simulation tool also created by MZA Associates. tempus provides the visual programming environment, the software architecture, and all the basic mechanisms that support model-building and simulation in the time domain. WaveTrain™ then adds a component library specifically for modeling optical systems and effects along with a handful of application specific graphical user interface (GUI) components. The later includes a tool for setting up AO system components (deformable mirror (DM) and wavefront sensor (WFS) via aogeom) and a tools for specifying the turbulence distribution along the optical propagation path (via PropConfig from ATMTools). These two MATLAB® based tools include additional utilities useful (and in some cases necessary) for additional functionality. For instance, the AO utilities include routines for computing the reconstructor matrices for the AO systems while ATMTools contains a variety of function for computing turbulence statistics over the atmospheric path and the mesh parameters for the numerical propagation.

1.1 Guide to the Quick Reference

To get a quick idea of what WaveTrain™ can do and how one can use it, start with our Quick Tour of WaveTrain™ (this should take less than 10 minutes). For a more detailed introduction (a few hours), work through the step-by-step tutorial in WaveTrain™ User Guide. After these introductory exercises, there will undoubtedly still be many details that are fuzzy to the new user. We suggest that the best procedure at that point is:

- Skim the rest of this Quick Reference and the User Guide to get a sense of the organization and contents
- Attempt construction of original simple systems, based on what you have learned from the ”Assembling” section and auxiliary tutorials in the User Guide
- As conceptual and procedural questions then arise, dive into the remaining detailed documentation chapters of the User Guide as needed

2. A Quick Tour of WaveTrain™

Modeling an optical propagation system in WaveTrain™ is essentially a four-step process. The steps are:

1. Assemble the WaveTrain™ model, by copying optical and processing components from the WaveTrain™ component libraries, and connecting the inputs and outputs of the components
2. Set the numerical parameters of all components, and the simulation timing parameters
3. Run the simulation
4. Inspect the WaveTrain™ simulation outputs, and post-process as needed

Steps (1)-(3), and parts of step (4), are carried out within WaveTrain™ 's visual programming environment. Arbitrary post-processing in step (4) requires the use of an auxiliary visualization and computation environment, such as MATLAB® .

2.1 Step 1 - Assembling and connecting the WaveTrain™ components

For example, suppose we want to look at the optical effects of propagation through atmospheric turbulence. We might assemble a basic model like shown in Fig. 2.1 of a simple point source imaging system. The window in Fig. 2.1 is called the TSE window (equivalently, the Block Diagram Editor or just System Editor), and is used for assembling the optical model from individual components. The displayed model consists of eight components (also called subsystems, or modules) connected together. All of the subsystems shown can be found in the WaveTrain™ component library, and all of the connections in this particular model represent optical interfaces. (Connections that correspond to electronic or general data signals are also available). Starting at the far right, we have a Point Source, a TransverseVelocity, an AtmoPath, another TransverseVelocity, a Telescope, and an IncomingSplitter, which splits the incoming light and sends part to a Camera and part to a SimpleFieldSensor. The PointSource represents an idealized point source, radiating uniformly in all directions. The two TransverseVelocities can be used to model source and detector transverse motion, and/or a true wind velocity. The AtmosphericPath module is a complex module that represents two processes: (a) the optical effects of turbulence using multiple discrete phase screens distributed along the propagation path, and (b) the diffractive propagation of light along the propagation path. By assigning screen motions, AtmosphericPath can also model transverse wind velocities that vary spatially along the propagation path. In between the phase screens the optical wavefronts are propagated using a two-step FFT propagator. The Telescope applies an aperture and a focus adjustment to the incoming light. The IncomingSplitter duplicates the incident beam and sends it out into two branches. The SimpleFieldSensor records the complex field - amplitude and phase - at the aperture plane. The Camera brings the light to a focal plane (using a Discrete Fourier Transform) and records the intensity pattern in that focal plane.

This relatively simple model can be assembled by a novice user in an hour or so - see our step by step tutorial - and it is not a toy; it can do serious calculations.

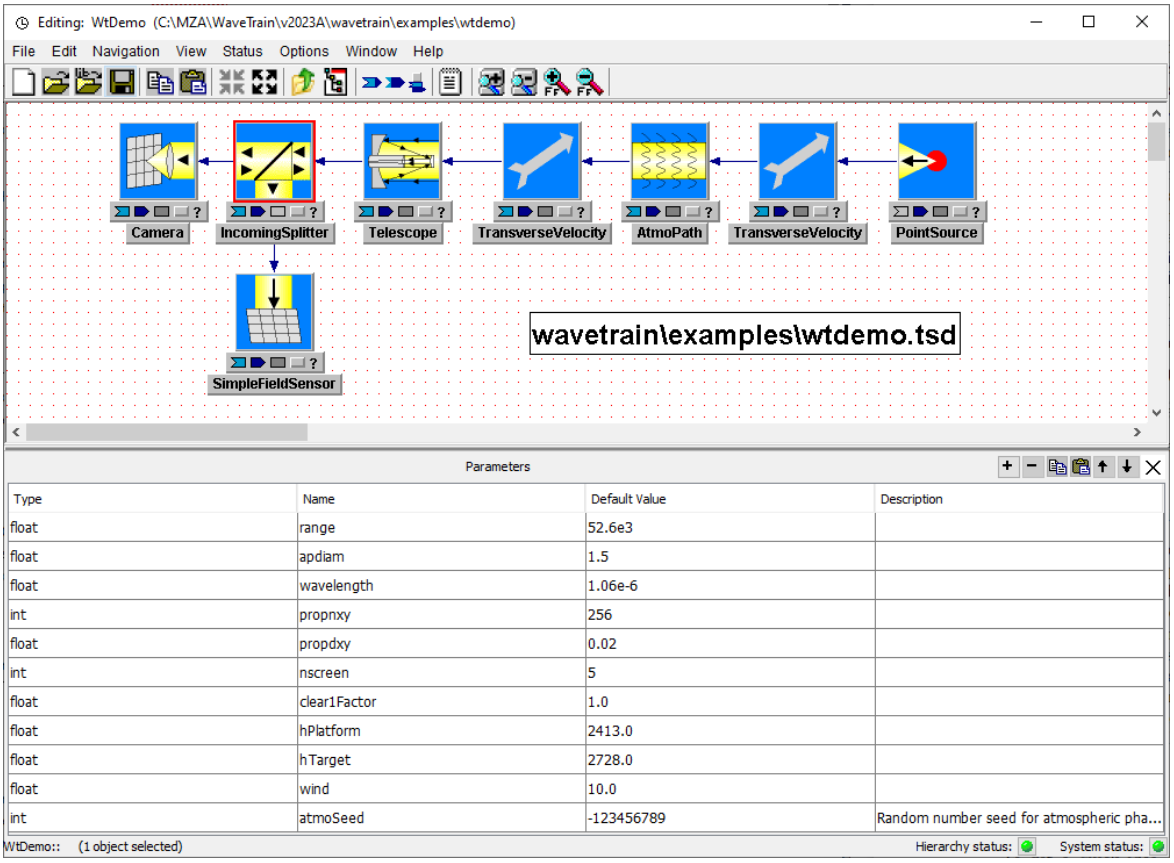

Figure 2.1: WaveTrain™ 's TSE, showing the WtDemo propagation model

2.2 Step 2 - Setting the numerical parameters of all components

After assembling and connecting the system components, the user must assign all the necessary parameter values. These include source powers, aperture diameters, propagation distances, propagation mesh dimensions, turbulence phase screen parameters, sensor spatial and timing parameters, etc. Setting all these parameter values is done partly in the TSE shown in Fig. 2.1 and partly in a second editor window, called the TSE. A sample runset for the WtDemo system is shown in Fig. 2.2.

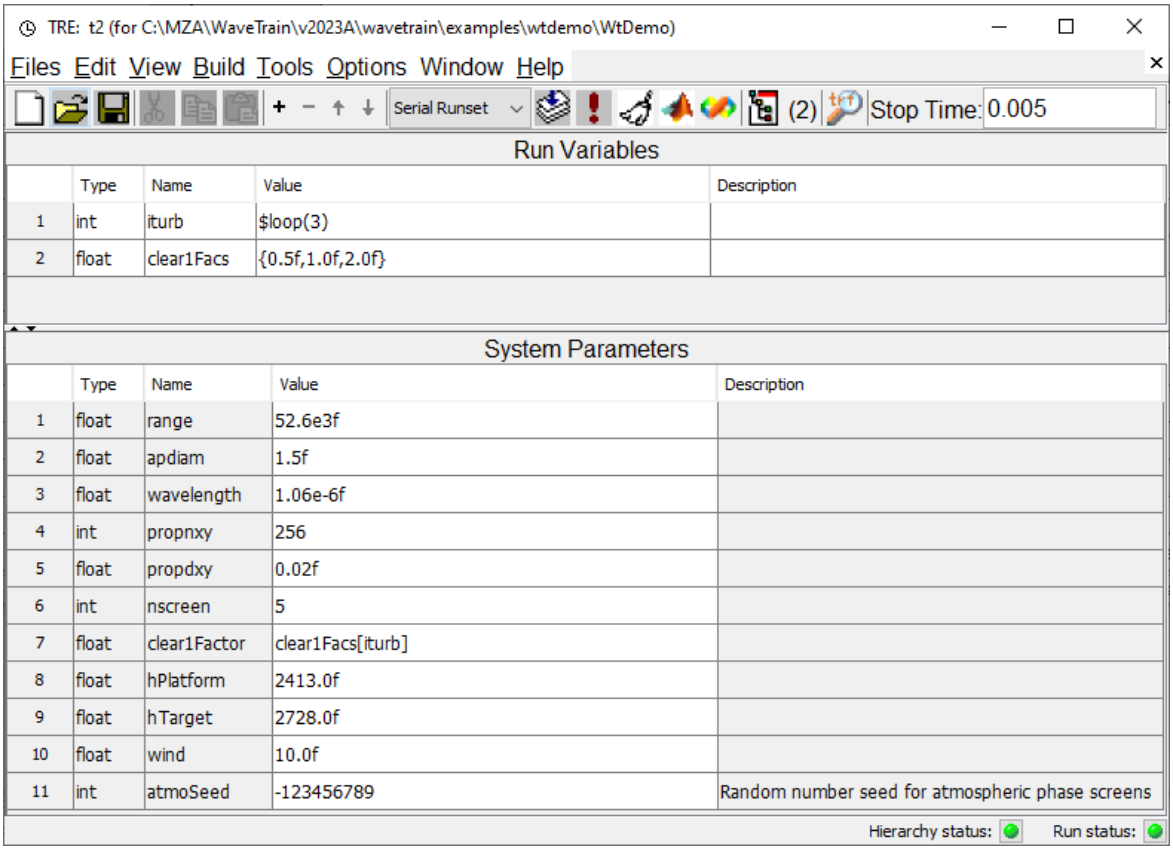

Figure 2.2: WaveTrain™ 's TSE, showing a runset for the WtDemo propagation model

2.3 Step 3 - Running the simulation

After specifying all the parameter values, the user initiates execution of the simulation with a few button clicks in the TSE. Upon completion of the execution, WaveTrain™ 's output - a combination of sensor images, complex field data, and/or electrical signal data - is stored to data files in a form that can be conveniently accessed for visualization and post-processing.

2.4 Step 4 - Inspecting and post-processing the WaveTrain™ outputs

Visualization and limited post-processing of outputs can be accomplished using the TrfView viewer that is part of WaveTrain™ suite. Typical outputs might be the instantaneous wave amplitude and phase maps in the "telescope1" pupil plane of the above WtDemo model. These particular outputs are based on the complex optical field sensed by the SimpleFieldSensor component in the model. Sample outputs as displayed in TrfView are shown in Fig. 2.3.

Completely general visualization and post-processing can be accomplished using WaveTrain™ 's MATLAB® interface. WaveTrain™ can be used with or without MATLAB® , but WaveTrain™ provides a full-featured MATLAB® interface. Not only can WaveTrain™ output data be conveniently imported into MATLAB® , but also complete WaveTrain™ system models

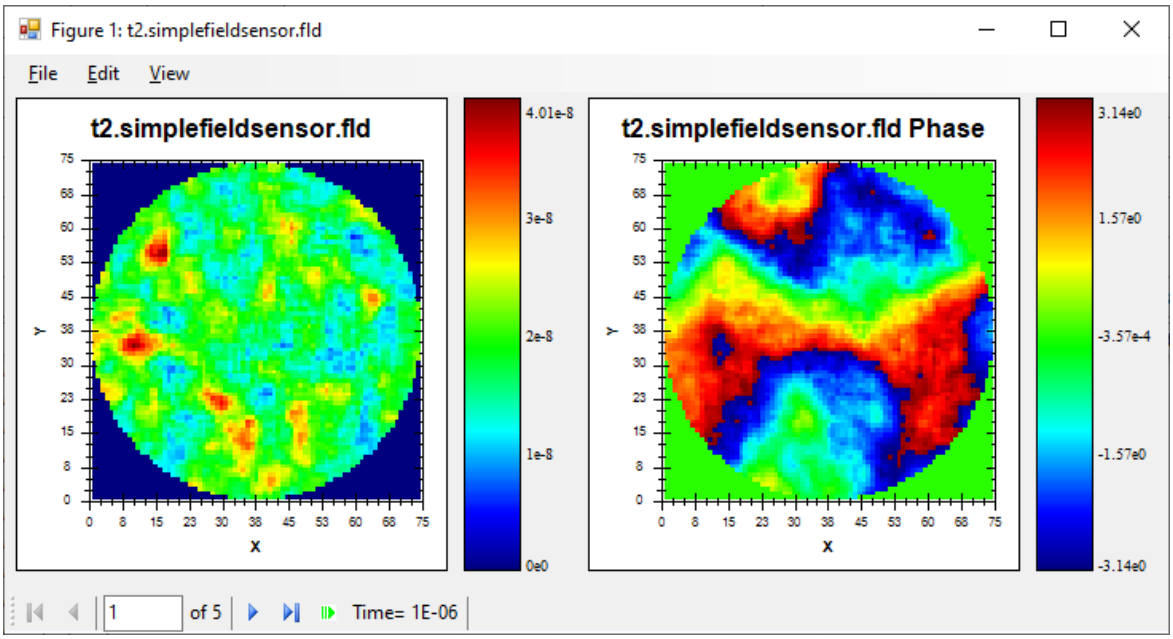

Figure 2.3: WaveTrain™ sample outputs - amplitude (left), and phase (right) maps in the telescope pupil of the WtDemo system, as displayed in WaveTrain™ 's TrfView viewer

can be created and executed from within MATLAB® , or converted into "S-functions" for use within Simulink, which is MATLAB® 's general purpose simulation environment. Additionally, MATLAB® m-files can be automatically converted into WaveTrain™ components, so that MATLAB® users have a simple way of creating custom WaveTrain™ components to supplement the WaveTrain™ library.

By varying the parameters of the WtDemo model illustrated in the Fig. 2.1 and Fig. 2.2 the user can investigate a wide variety of issues fundamental to the problem of optical propagation through turbulence. These issues include the statistics characterizing the spatial and temporal variation of the amplitude and phase effects, in both pupil and image planes of a receiver, and the way those statistics changes when the distribution of turbulence along the path is changed.

2.5 Auxiliary tools for defining input parameter specifications

In addition to the fundamental TSE and TRE, the WaveTrain™ suite includes several other GUIs. The purpose of these auxiliary tools is to assist in setting the numerical inputs of some of the WaveTrain™ components, particularly the more complicated modules that deal with atmospheric turbulence and AO components. For example, there is a MATLAB® GUI for setting up deformable mirror and wavefront sensor geometries. Also, there is a MATLAB® GUI that facilitates turbulence strength and atmospheric specifications and allows the evaluation of certain analytical formulas that estimate key integrated-turbulence quantities (such as scintillation variance and Fried's r0).

2.6 Scope and development of WaveTrain™

WaveTrain™ is designed to be useful to scientists, engineers, teachers, and students engaged in the design and development of advanced optical systems, or in the study of propagation through turbulence and associated AO systems. It is equally well-suited for simple models, like the example shown above, for intermediate complexity models such as are typically used in concept exploration, and for the kind of detailed engineering models necessary for accurate performance prediction, design refinement, and trouble-shooting. We are continually adding new components and features, and making improvements, often in response to customer feedback. If you need or want a feature that we don't yet offer, please let us know. Alternatively, WaveTrain™ is designed to be extensible, so you can create your own components to supplement the WaveTrain™ library, as described in the chapter on creating your own WaveTrain™ components.

References

[1] J. W. Goodman, *Statistical Optics*. New York: Wiley, 1985.

[2] V. I. Tatarskii, *Wave Propagation in a Turbulent Medium*. New York: Dover, 1961.

[3] A. Ishimaru, *Wave Propagation and Scattering in Random Media*, vol. 2. New York: Academic Press, 1978.

[4] R. K. Tyson, *Principles of Adaptive Optics*. Boston: Academic Press, 3rd ed., 2010.

[5] M. C. Roggemann and B. Welsh, *Imaging Through Turbulence*. Boca Raton: CRC Press, 1996.

[6] J. Hardy, *Adaptive Optics for Astronomical Telescopes*. Oxford: Oxford University Press, 1998.

[7] L. C. Andrews and R. L. Phillips, *Laser Beam Propagation through Random Media*. Bellingham: SPIE Press, 2nd ed., 2005.

[8] R. R. Beland, "Propagation through atmospheric optical turbulence," in *The Infrared and Electo-Optical Systems Handbook. Vol. 2, Atmospheric Propagation of Radiation* (F. G. Smith, ed.), pp. 157–232, Bellingham: ERIM and SPIE Optical Engineering Press, 1993.

[9] J. D. Schmidt, *Numerical Simulation of Optical Wave Propagation with Examples in MATLAB*. Bellingham: SPIE Press, 2010.

AO	adaptive optics
DM	deformable mirror
GUI	graphical user interface
RHEL	Red Hat Enterprise Linux
TRE	Tempus Runset Editor
TSE	Tempus System Editor
WFS	wavefront sensor
WTC	Concourse