Quarter-Wave Plate Mirror

Educational Tutorial

Why Educational Tutorials?

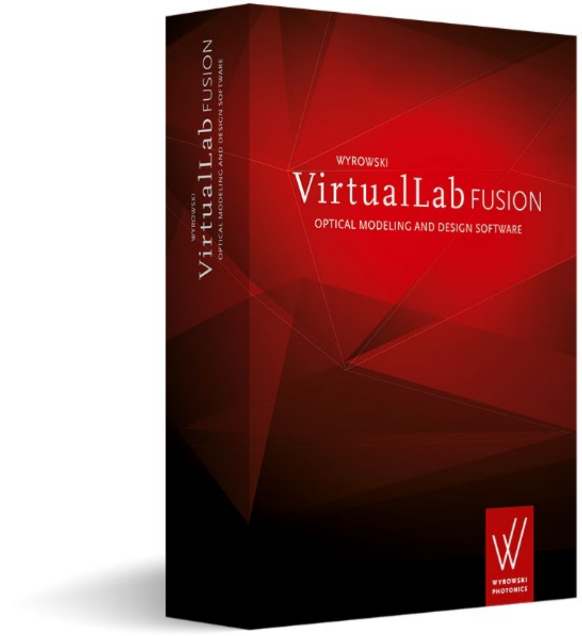
Learn software and physics — enjoyably and effectively.

Our hands-on tutorials are designed to make learning *VirtualLab Fusion* both engaging and efficient.

The fastest way to learn something new is by doing it yourself, especially when guided by a good example. These tutorials offer a practical, step-by-step approach that works even with the **trial license**.

You'll not only become familiar with the software, but also gain valuable insights into the underlying physics, making the tutorials ideal for **self-study, teaching**, and as a starting point for **engineers** and **scientists** exploring the software's possibilities.

Real, working examples bring abstract concepts to life far more effectively than just reading about features in a manual. And by the end, you'll feel confident enough to start building your own simulations.



Educational Tutorial #2 – Quarter-Wave Plate Mirror

Advancing Your Software Skills

In this tutorial, we'll design a quarter-wave plate (*QWP*) mirror, building on the concepts from the first educational tutorial on the Fabry-Pérot Interferometer.

We'll move through the basic steps more quickly, assuming you're already familiar with:

- building *Optical Setups*,
- using the *Light Path Finder*,
- and the *Parameter Run* tool.

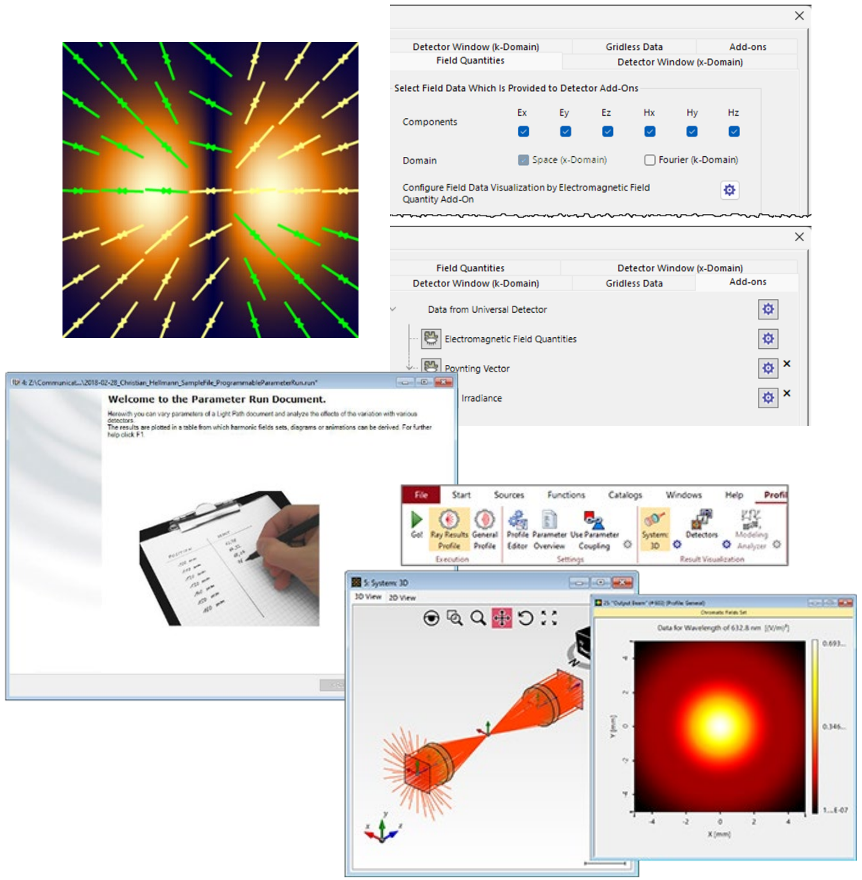
If you encounter any difficulties, we strongly recommend revisiting the first tutorial for a clearer understanding.

Educational Tutorial #2 – Quarter-Wave Plate Mirror

Main New Concepts in Tutorial #2:

- *Polarization Ellipses*
- Programmable *Detector Add-on*
- 2D *Parameter Run*
- *Parameter Optimization*

By the end, you'll deepen your understanding and feel more confident using the software for your own simulations.

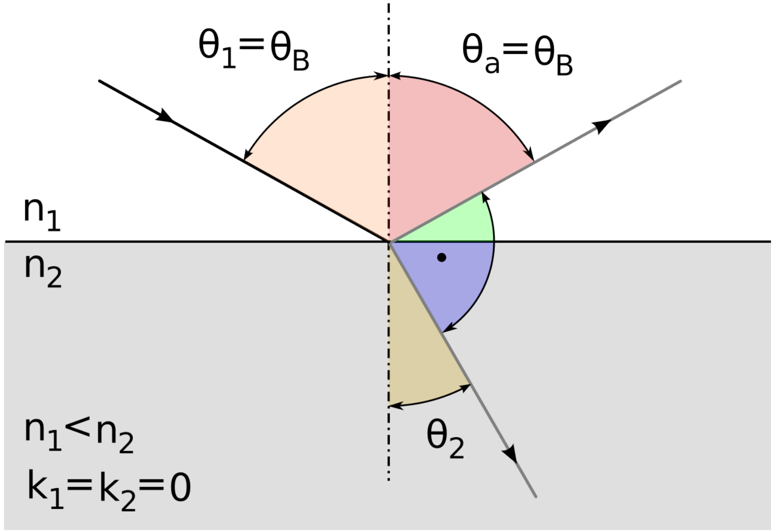


Educational Tutorial #2 – Quarter-Wave Plate Mirror

Physics background

At the interface of two media with different refractive indices, *p*- and *s*-polarized light exhibit different reflection coefficients. This effect is most pronounced at the *Brewster Angle*, where *p*-polarized light is fully transmitted, leaving the reflected light purely *s*-polarized.

We will leverage this principle to design a mirror that transforms linearly polarized light into circularly polarized light, effectively acting as a quarter-wave plate, without requiring birefringent materials.



For non-absorbing media, the Brewster angle is given by

$$\theta_B = \tan^{-1}(n_2/n_1)$$

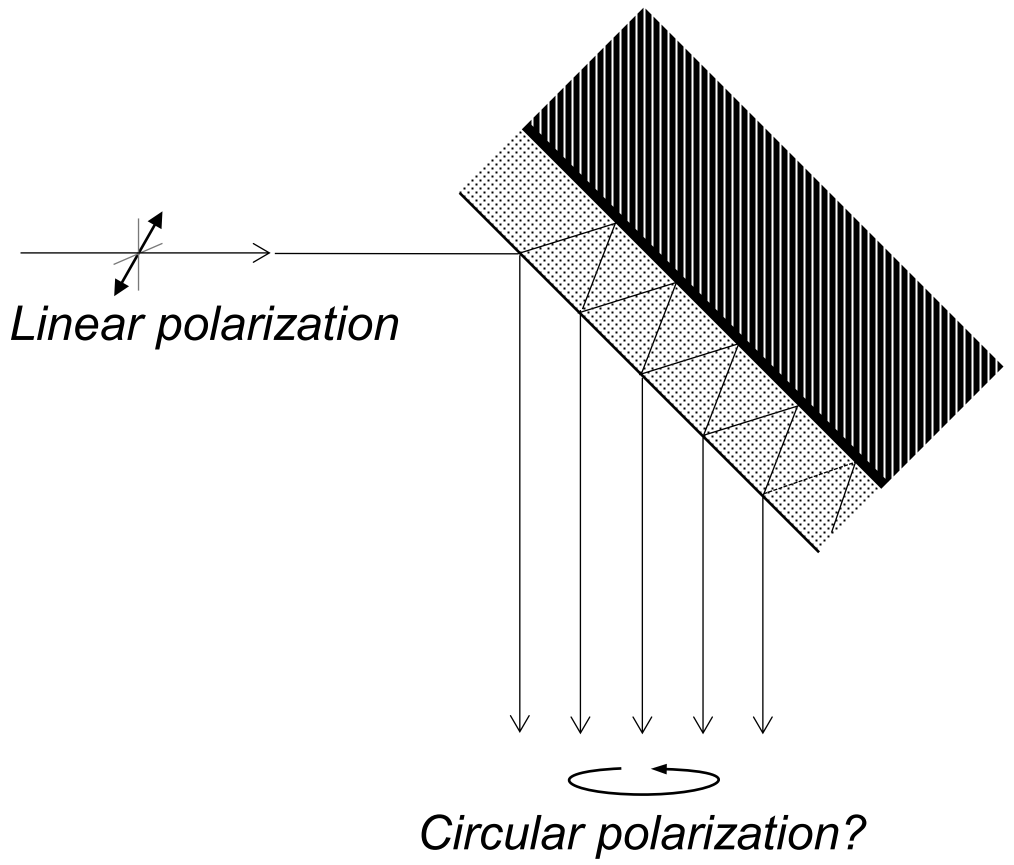
Overview

The *QWP* mirror will consist of an ideal reflector coated with a thin, glass-like layer. Partial reflections at the front surface create different paths that all eventually reach the detector.

At the air-glass interface, the reflection coefficients differ for *s*- and *p*-polarized light, giving their paths different “weights” or strengths. As a result, both polarizations experience different phase delays on average. However, the total amount of *s*- and *p*-polarized light remains the same.

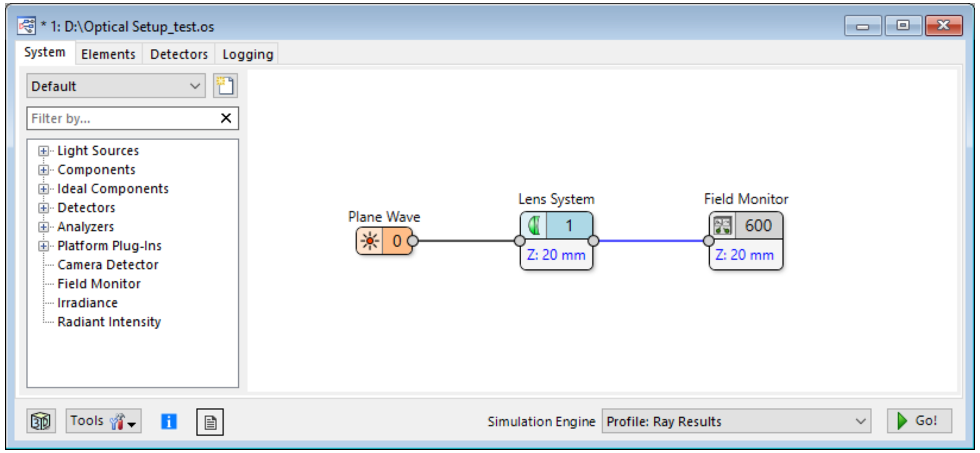
Circularly polarized light requires *s*- and *p*-polarized waves of equal amplitude with a 90° phase difference. To achieve this, the linear polarization direction of the incident light must be at 45° to the reflection plane, ensuring equal *s*- and *p*-components, while the coating provides the necessary 90° phase shift.

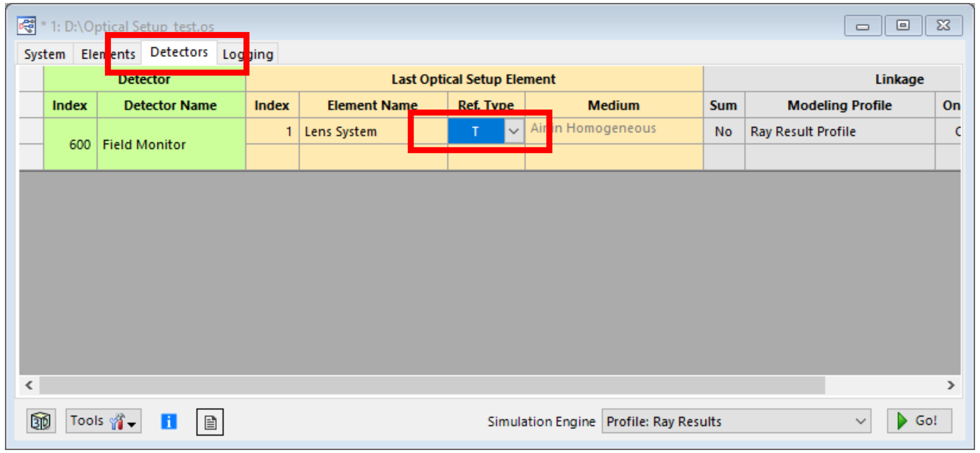
In the first test, light hits the mirror at a 45° angle, and the coating thickness is adjusted. But in this setup, it’s not possible to create a 90° phase shift. A more detailed study shows that the phase shift can be achieved at larger angles of incidence.



Optical Setup

- Start a new *Optical Setup* in *VLF*
- Add a *Plane Wave* source, a *Lens System*, and a *Field Monitor*
- Set all *z*-distances to 20 mm

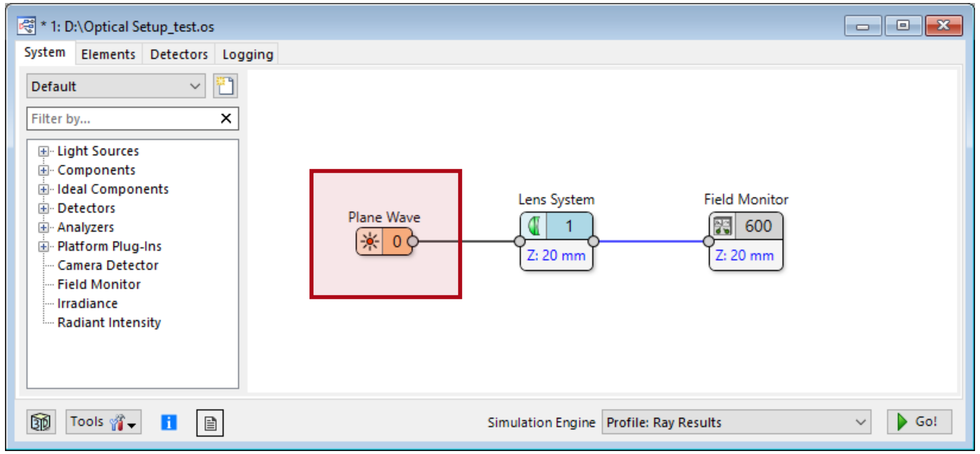




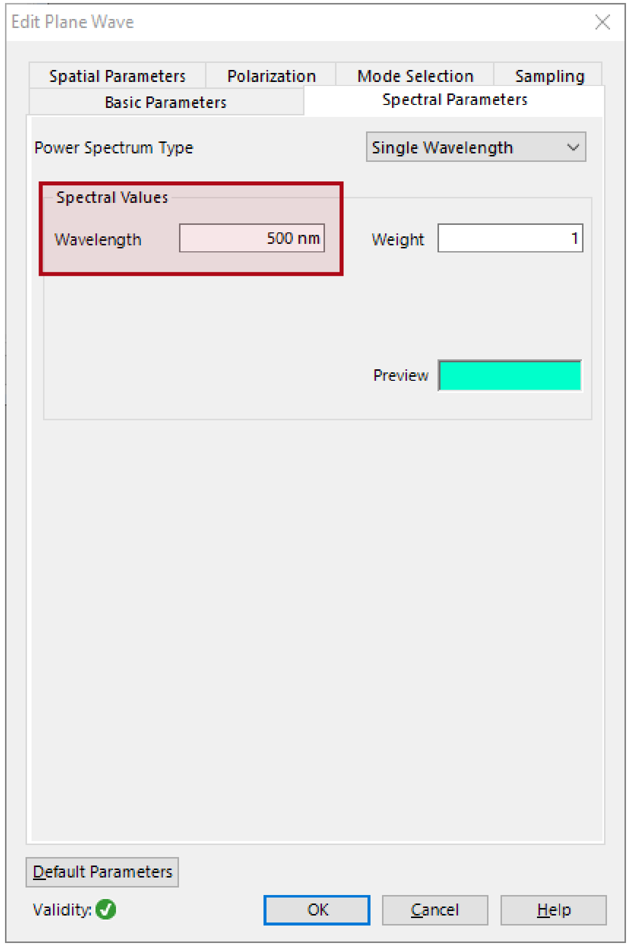
- Go to the *Detectors* tab
- Set *Reference Type from (T)* to *Reflective (R)*
  - ➢ The detector will auto-align along the specular reflection axis of the lens system

Optical Setup – Source

- Double click on the *Plane Wave* icon

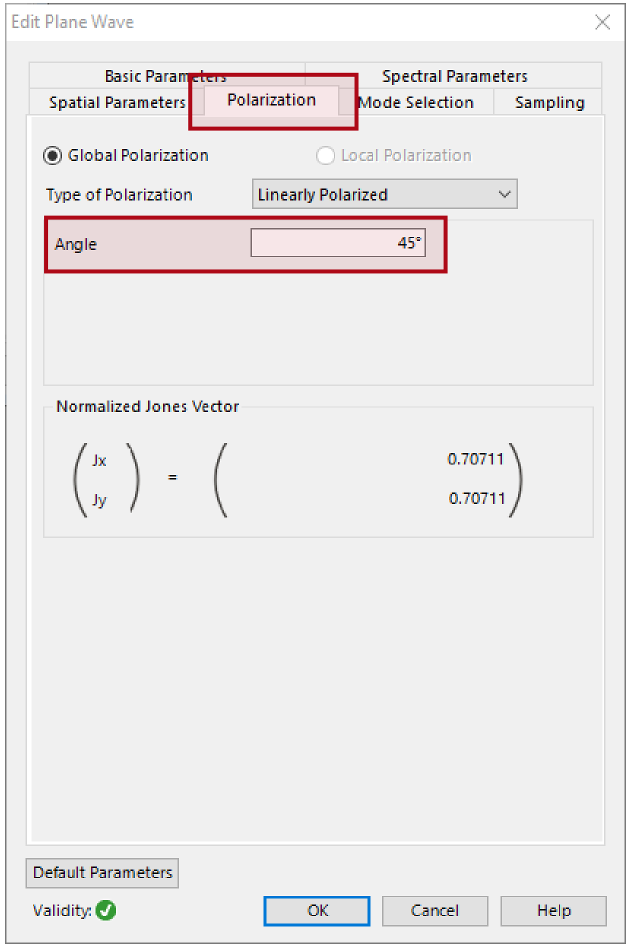


- Set *Wavelength* to 500 nm

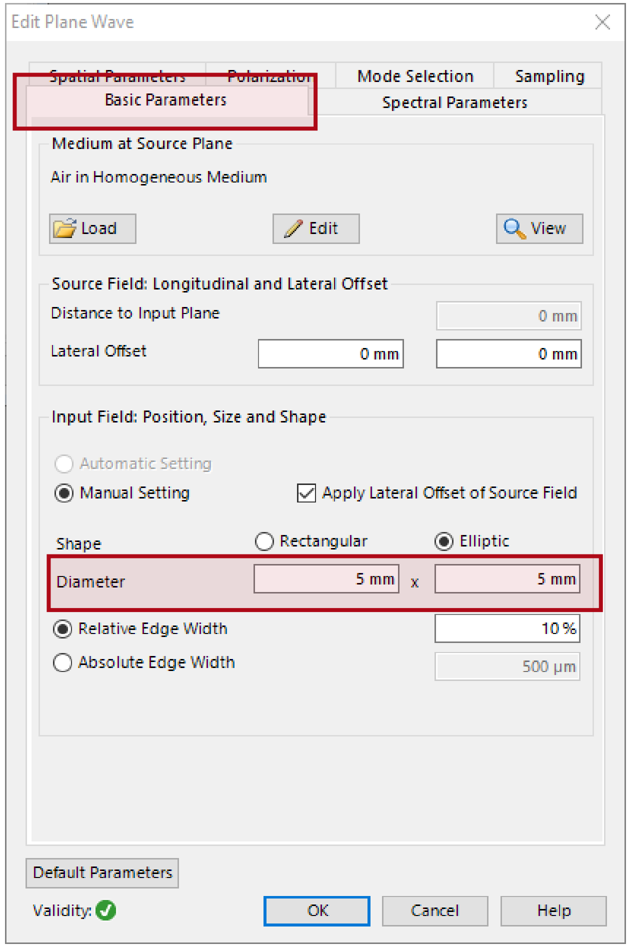


Optical Setup – Source

- Set the linear polarization to 45°



- Set *Diameter* to 5 mm x 5 mm

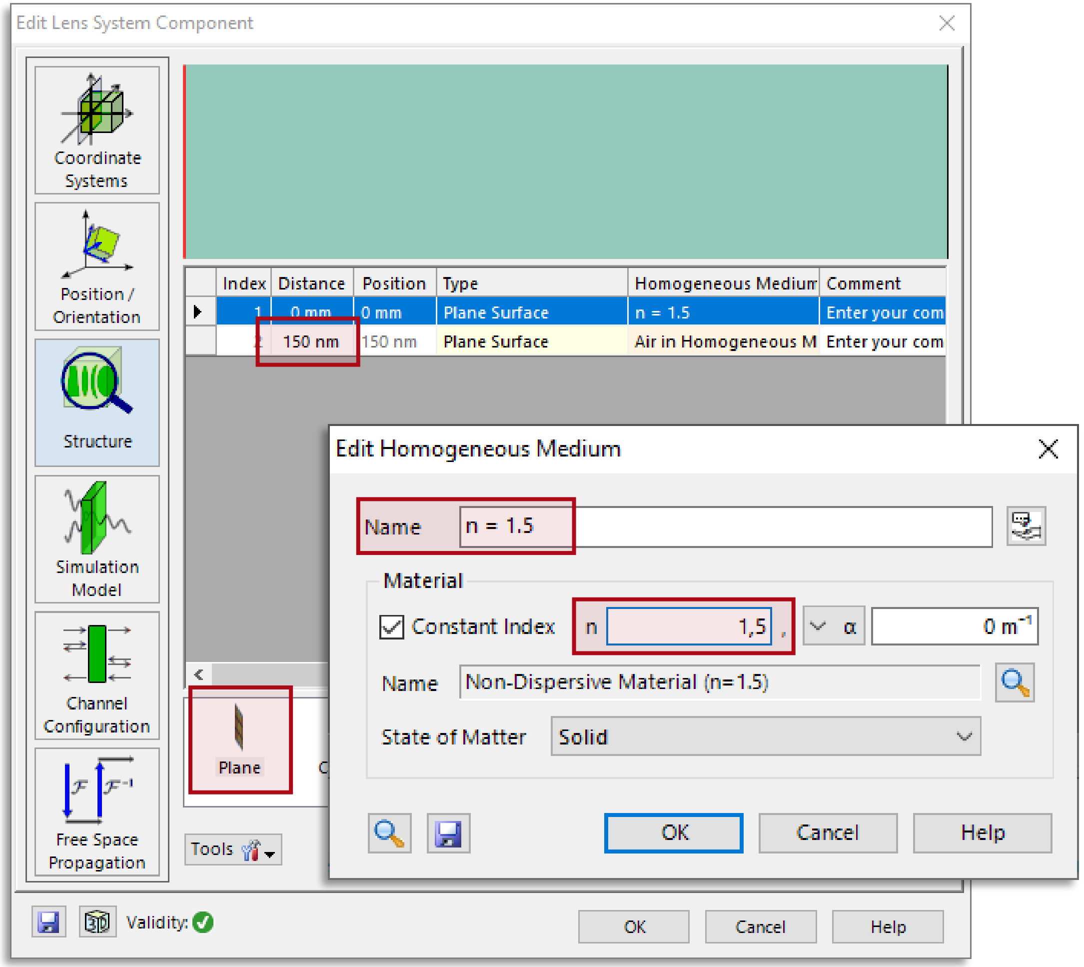


Optical Setup – Mirror Coating

The mirror consists of two surfaces with a medium between them.

- Edit *Lens System*
- Drag & drop two plane surfaces into the system.
- Edit the *Homogeneous Medium* behind the first surface
- Change the *Name*
- Choose *Defined by Constant Refractive Index* and set the value to 1.5
- Set the distance of the second surface to the first surface to 150 nm

This creates a 150 nm thick layer of material with n = 1.5 between the surfaces.

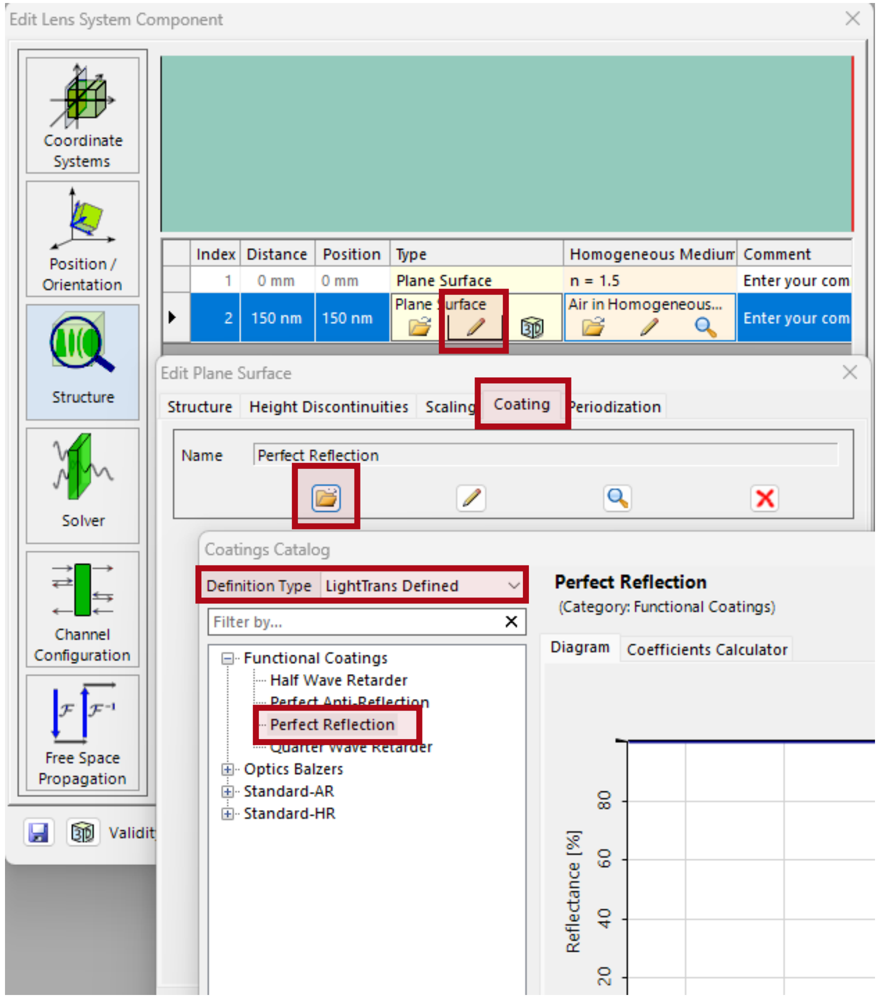


Optical Setup – Mirror Substrate

Make the mirror substrate perfectly reflective:

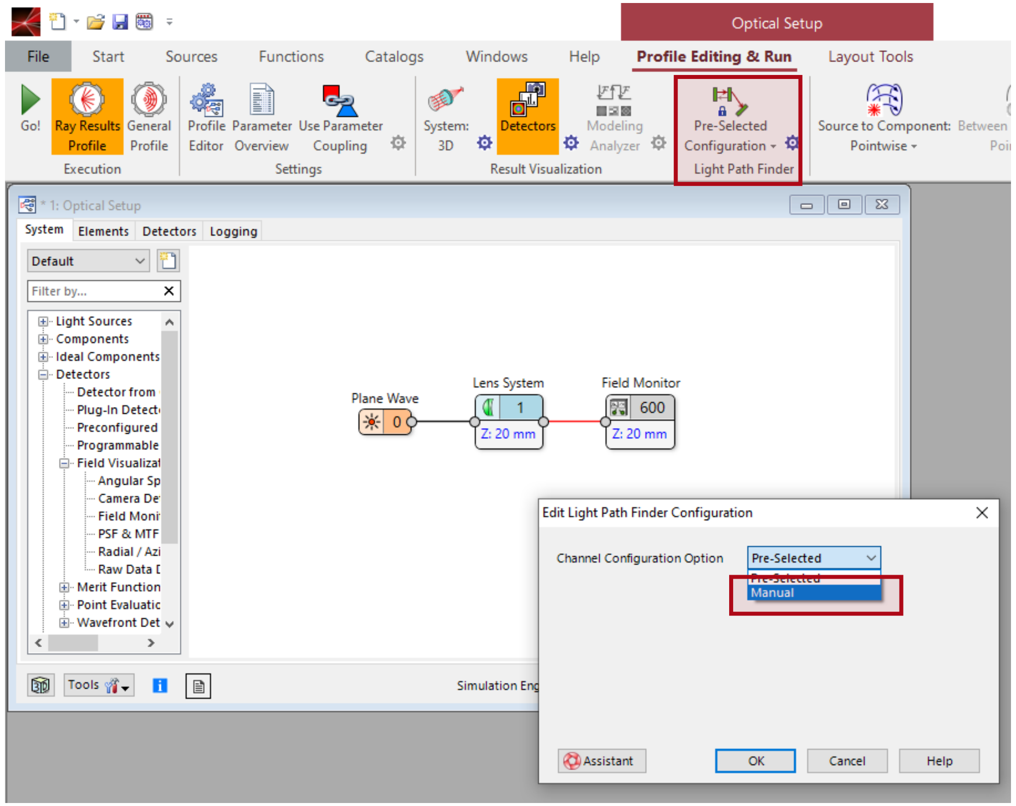
- Edit the second *Plane Surface*
- Go to the *Coating* tab
- Click the *folder* icon to load a coating
- Set *Definition Type* to *LightTrans Defined*
- Under *Functional Coatings*, select *Perfect* Reflection (Click the *pencil* icon to view its properties, if desired)
- Click OK three times to apply and close all dialogs

Note that the medium behind the second surface has no further effect.

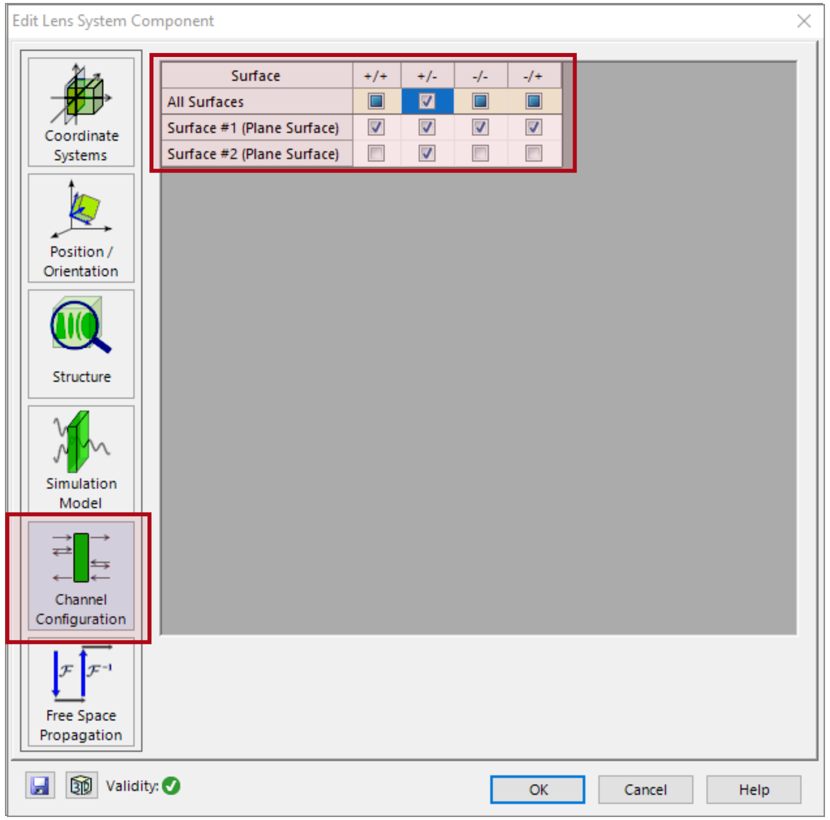


Optical Setup – Channel Configuration

- Click the *cogwheel* ( ) icon in the *Light Path Finder*
- Set *Channel Configuration Option* to *Manual*
- Click *OK*

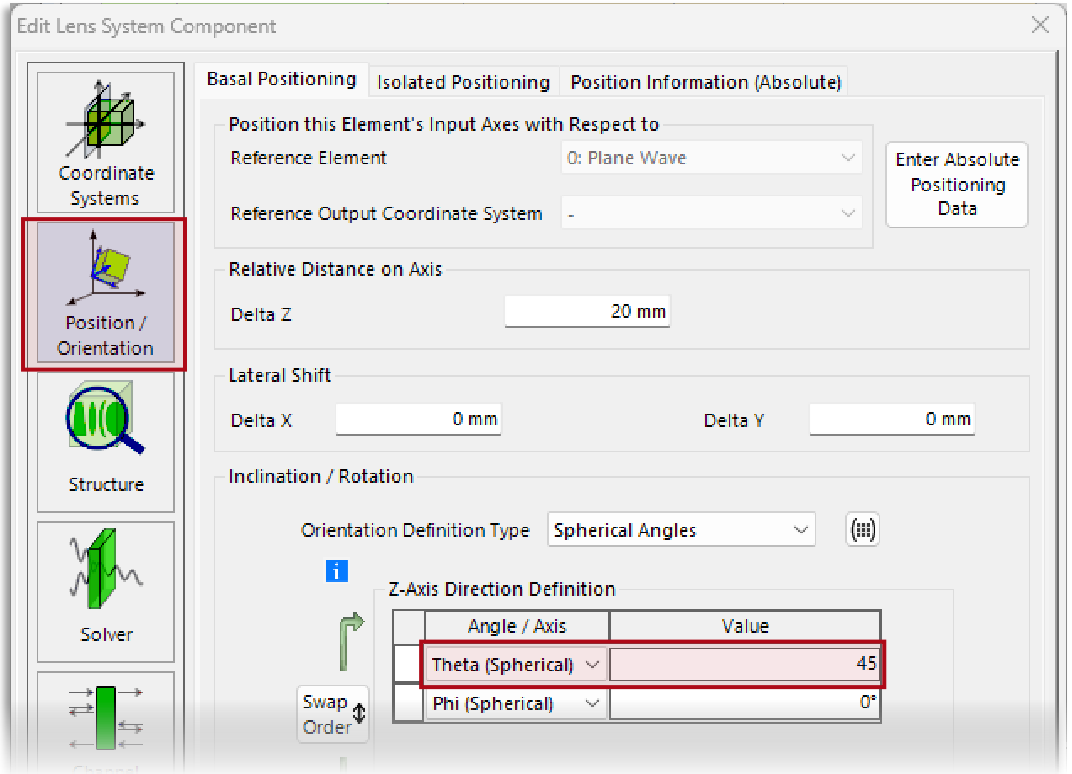


- Edit the *Lens System*
- Select *Channel Configuration*
- For *Surface # 1*, select all channels
- For *Surface # 2*, select the +/- channel

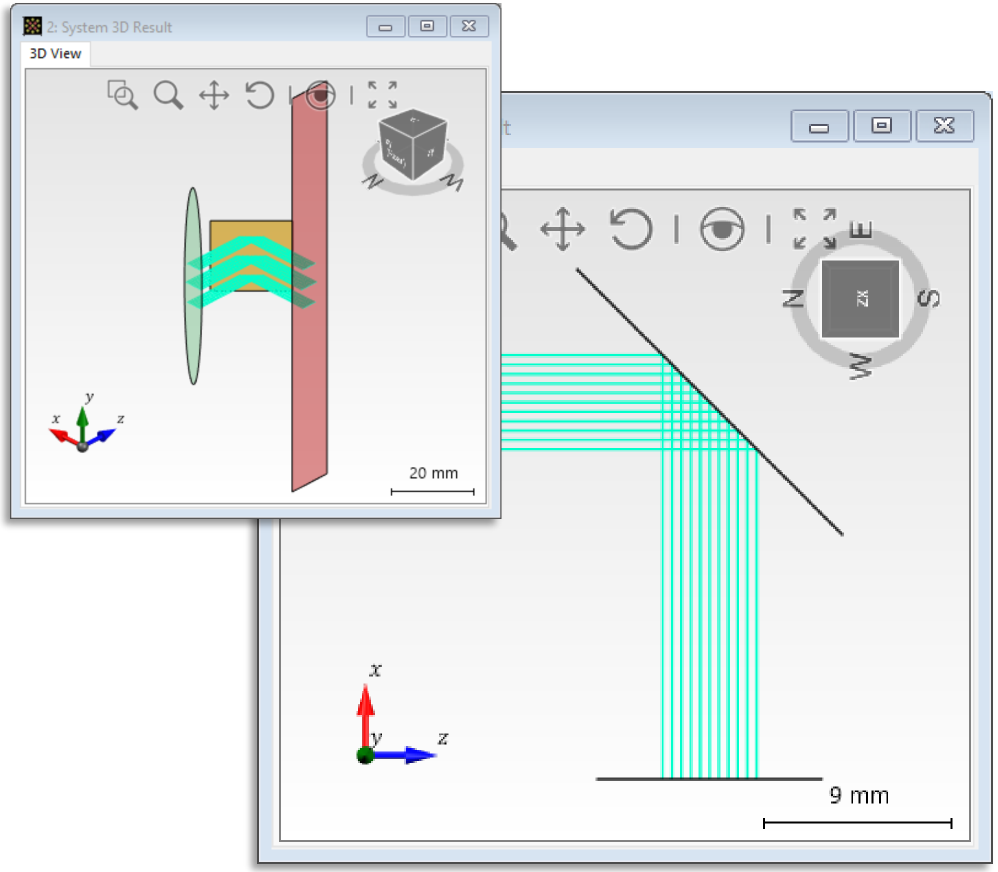


Set Orientation & Run Ray Profile

- In *Position/Orientation*, set Theta to 45°
- Click *OK* to apply all changes

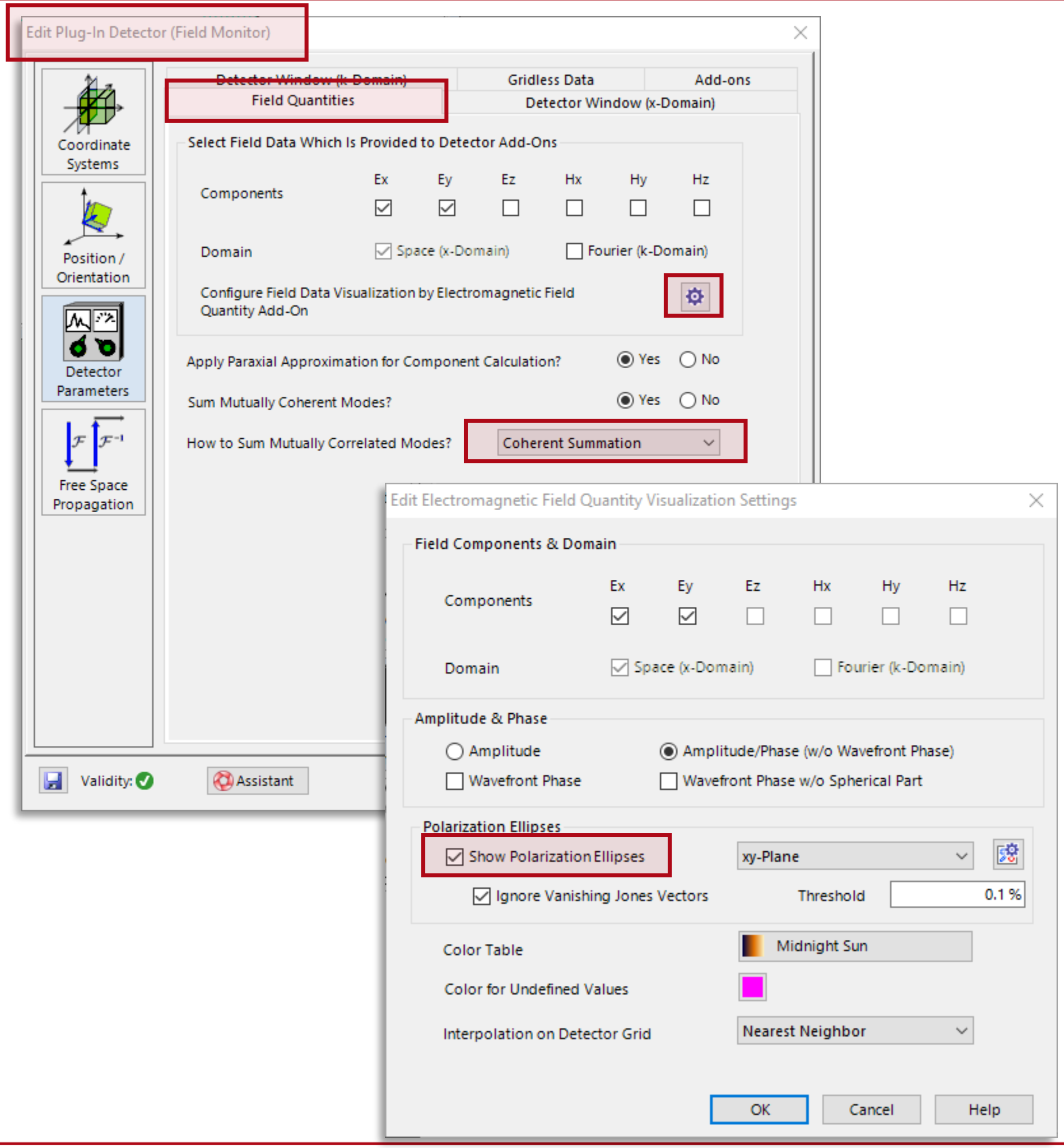


- Run the simulation in *Ray Profile, System: 3D*, to ensure the setup matches the expected configuration



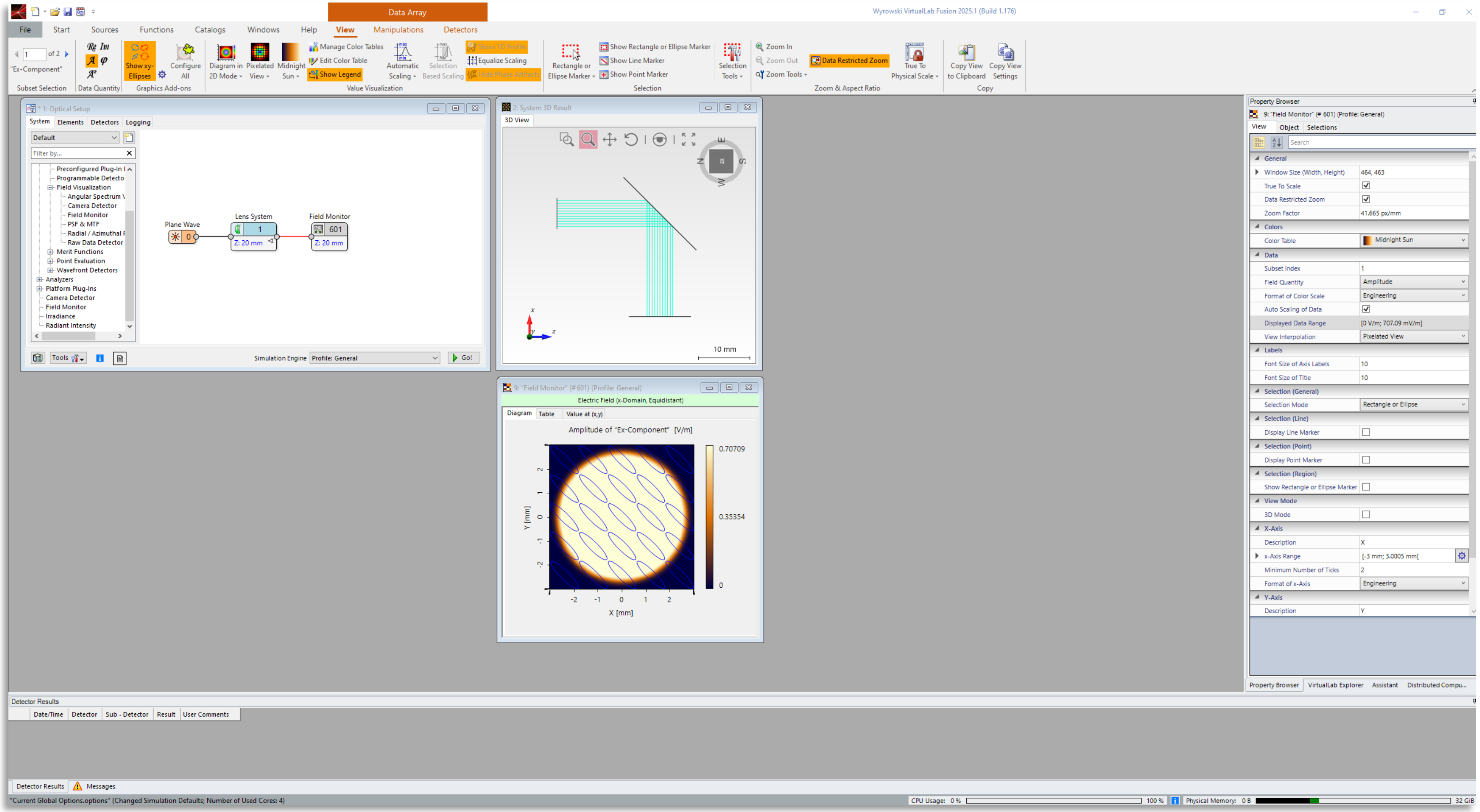
General Profile – Set up the Detector

- Switch to *General Profile*
- Edit the *Field Monitor*
- Go to the *Field Quantities* tab
- Make sure *Coherent Summation* is selected
- Click the *cogwheel* ( ) icon
- Check the box *Show Polarization Ellipses*
- Close all dialogs with *OK*
- Run the simulation



General Profile – Set up the Detector

Your screen should look like this:



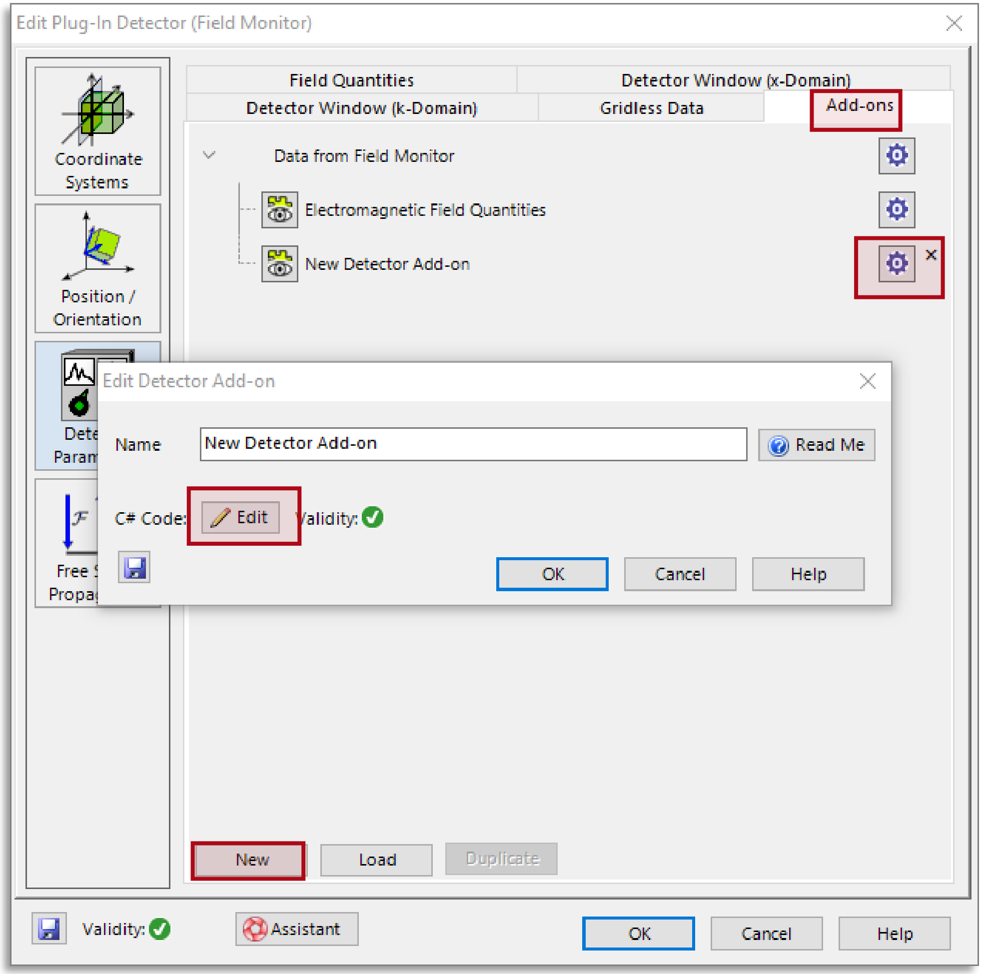
Create a new Detector Add-on

VirtualLab Fusion provides the amplitudes and phases of *Ex* and *Ey* at the detector.

Our goal is to calculate the phase difference between them and adjust the setup parameters so that this difference becomes ±90°.

For demonstration purposes, we will use the optimization routine, which requires a single-value merit function – in this case, the phase offset. To enable this, we'll program a simple *Detector Add-on* that outputs the required value:

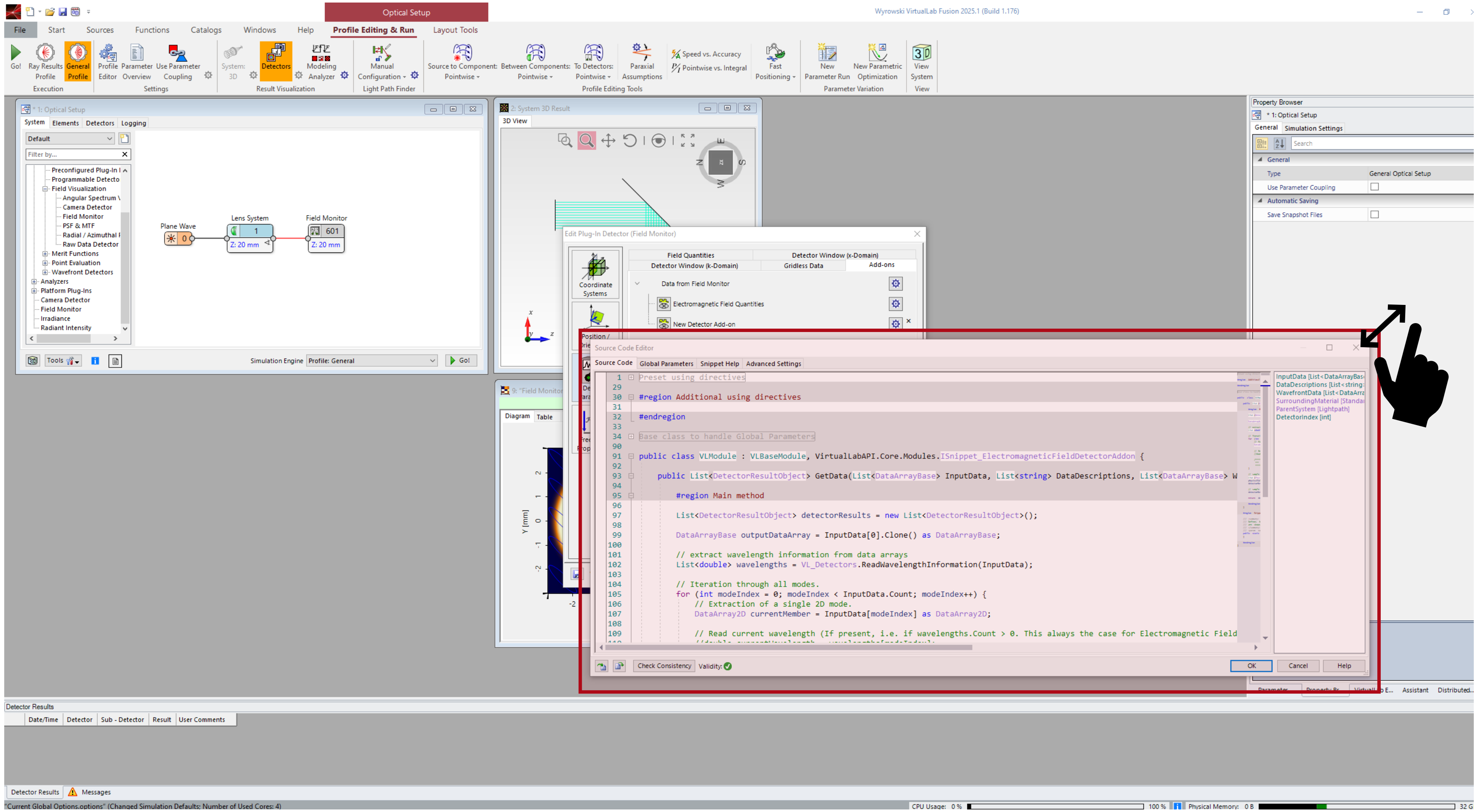
- Edit the *Field Monitor*
- Go to the *Add-ons* tab
- Click *New*
- Click the *cogwheel* icon
- Select *Edit* to open the C# code editor



Study the Snippet

After clicking *Edit*, you'll see the code for a functional add-on. Before making any changes, let’s review the key parts of the existing code relevant to our task.

Scale the Source Code Editor to a size that suits you.



Study the Snippet

Line 99 and 102 are comments and do not affect the code.

Line 100 reads the wavelengths used in the simulation into a local list. Since we won’t use wavelengths to calculate the phase offset, we will delete this line later, although leaving it doesn’t cause any issues.

```
99   // extract wavelength information from data arrays
100  List<double> wavelengths = VL_Detectors.ReadWavelengthInformation(InputData);
101
102  // Iteration through all modes.
103  for (int modeIndex = 0; modeIndex < InputData.Count; modeIndex++) {
```

At line 103, the code starts looping through all modes. The loop code ends with the closing bracket at line 113.

```
113  }
```

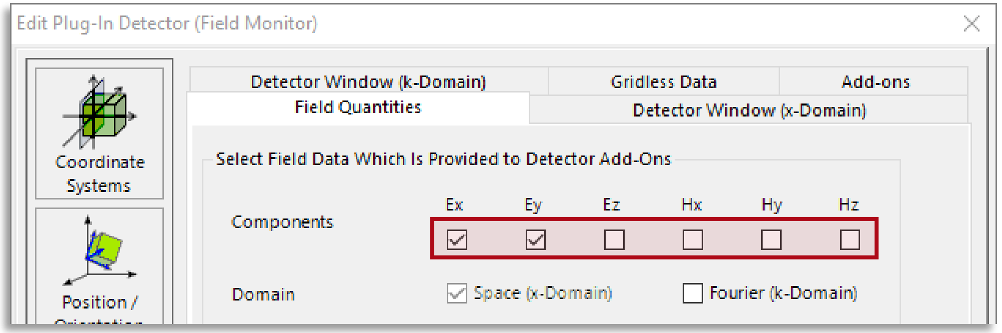
Modes are different entries in the electromagnetic field data set. In this simulation:

- We use only one wavelength
- We have chosen to sum coherent modes, meaning we combine all possible light paths to the detector

Since this results in just one mode, we could remove the loop—but we’ll keep it for clarity.

Study the Snippet – Part 2

At line 105, the current mode's electromagnetic field is read into *currentMember,* a 2D array representing the detector's *x* and *y* dimensions. In this simulation, *currentMember* holds the complex values of *Ex* and *Ey*, the only components selected for this detector.




```
104        // Extraction of a single 2D mode.
105        DataArray2D currentMember = InputData[modeIndex] as DataArray2D;
```

Line 117 generates a list of a value, its unit, and its name. Line 118 stores it in the output data of the detector.

This type of one-dimensional information will appear in the *Detector Results* window.

```
115     // sample detector output for physical values
116     List<PhysicalValue> physicalValues = new List<PhysicalValue>();
117     physicalValues.Add(new PhysicalValue(1, PhysicalProperty.Length, "My Detector Result"));
118     detectorResults.Add(VL_Detectors.CreateDetectorResult(physicalValues));
```

Study the Snippet – Part 3

Line 121 adds the contents of *outputDataArray* to the detector's output data. This two-dimensional information will be displayed in a result window.

```
120         // sample detector output for documents
121         detectorResults.Add(VL_Detectors.CreateDetectorResult(outputDataArray, "My Detector Result"));
```

- Close the snippet by clicking *Cancel*, to ensure no changes are made to the code
- Close the *Edit Field Monitor* dialogue with *OK*, to activate *New Detector Add-on*
- Run the simulation

The *Detector Results* window will display the name of the add-on along with the information defined in line 117 of the snippet. Tip: You can find *Detector Results* next to the *Messages* tab at the bottom of the GUI.

Detector Results

	Date/Time	Detector	Sub - Detector	Result	User Comments
1	2025-07-16 10:11:18	"Field Monitor" (# 601): New Detector Add-on (Profile: General)	My Physical Value Result	1 m	

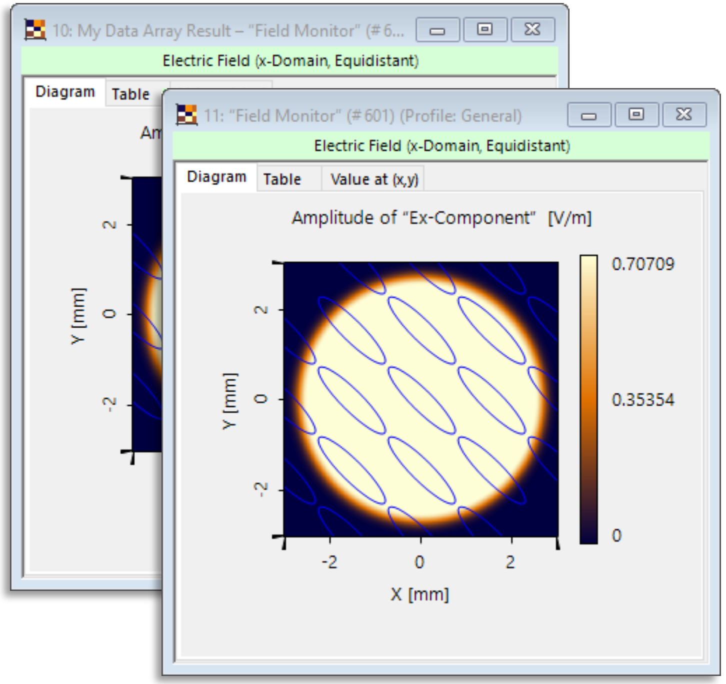
Study the Snippet – Part 3

Two result windows will appear:

one named *Field Monitor*

and the other *My Data Array Result*

The latter appears as a consequence of line 121 in the code of the new add-on. Line 97 clarifies why both windows display identical data, considering the snippet contains no code that modifies the output.




```
97        DataArrayBase outputDataArray = InputData[0].Clone() as DataArrayBase;
```

Update the Snippet

```
#region Main method

List<DetectorResultObject> detectorResults = new List<DetectorResultObject>();

DataArrayBase outputDataArray = InputData[0].Clone() as DataArrayBase;

// extract wavelength information from data arrays
List<double> wavelengths = VL_Detectors.ReadWavelengthInformation(InputData);

// Iteration through all modes.
for (int modeIndex = 0; modeIndex < InputData.Count; modeIndex++) {
    // Extraction of a single 2D mode.
    DataArray2D currentMember = InputData[modeIndex] as DataArray2D;

    // Read current wavelength (If present, i.e. if wavelengths.Count > 0. This
    //double currentWavelength = wavelengths[modeIndex];

    /*****************************************************
     *** DO ALL EVALUATION ON THE CURRENT SUBSET HERE ***
     *****************************************************/
}

// sample detector output for physical values
List<PhysicalValue> physicalValues = new List<PhysicalValue>();
physicalValues.Add(new PhysicalValue(1, PhysicalProperty.Length, "My Detector Re
detectorResults.Add(VL_Detectors.CreateDetectorResult(physicalValues));

// sample detector output for documents
detectorResults.Add(VL_Detectors.CreateDetectorResult(outputDataArray, "My Detec

return detectorResults;

#endregion
```

```
#region Main method

List<DetectorResultObject> detectorResults = new List<DetectorResultObject>();

DataArrayBase outputDataArray = InputData[0].Clone() as DataArrayBase;

// declare and initialize local variables
double phase_x = 0;
double phase_y = 0;
double phase = 0;

// Iteration through all modes.
for (int modeIndex = 0; modeIndex < InputData.Count; modeIndex++) {
    // Extraction of a single 2D mode.
    DataArray2D currentMember = InputData[modeIndex] as DataArray2D;

    phase_x = currentMember.Data[0][256, 256].Arg(); // phase of Ex[256, 256]
    phase_y = currentMember.Data[1][256, 256].Arg(); // phase of Ey[256, 256]
    phase = phase_x - phase_y;                       // the phase difference

    while (phase > Math.PI)            //   This block ensures
        phase -= 2 * Math.PI;          //  the phase difference
    while (phase < -Math.PI)           //       is within
        phase += 2 * Math.PI;          //    the range -π to π
}

// sample detector output for physical values
List<PhysicalValue> physicalValues = new List<PhysicalValue>();
physicalValues.Add(new PhysicalValue(phase, PhysicalProperty.AngleDeg, "Offset"));
detectorResults.Add(VL_Detectors.CreateDetectorResult(physicalValues));

return detectorResults;

#endregion
```

Finish Editing and Go!

- Make all modifications to the snippet
- At the bottom-left corner of the *Source Code Editor*, click *Check Consistency* to find syntax errors (e.g. missing semicolons)
- Click *OK*, then rename the add-on to *Phase Offset Add-on*
- Click *OK* twice to save and close all dialogs
- Clear the *Detector Results* window (Tip: Right-click)
- Run the simulation.
- Inspect the *Detector Results* window

Detector Results

	Date/Time	Detector	Sub - Detector	Result	User Comments
1	2025-07-16 10:28:38	"Field Monitor" (# 601): Phae Offset Add-on (Profile: General)	Offset	-154.65°	

Finish Editing and Go!

Summary of Modifications:

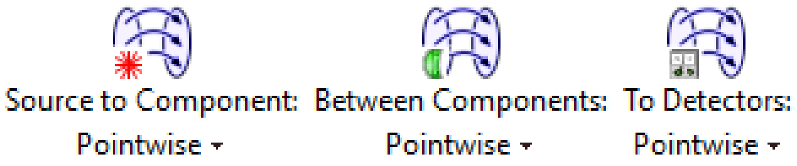
- Delete reading wavelength information
- Declare new variables for storing the phases of *Ex*, *Ey* and the phase offset
- Inside the mode-loop, remove the green comments and replace them with the code shown in lines 109 to 116.
- Use indices [256, 256] since these fall within the flat part of the plane wave
- In line 122 (*PhysicalValues.Add*) replace:
    - *1* with *phase*
    - *Length* with *AngleDeg*
    - *My Detector Result* with *Offset*
- Delete the code responsible for the result window

Assess Diffraction and the Paraxial Approximation

Diffraction

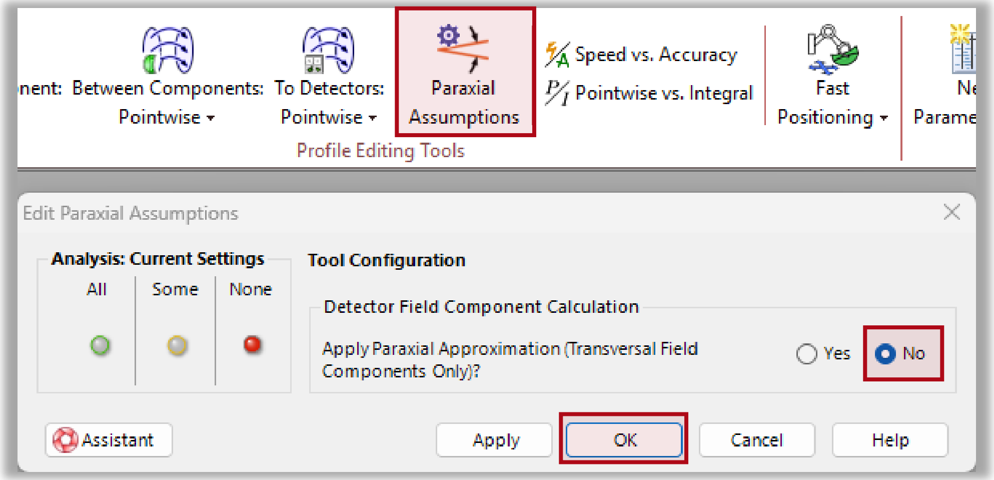
In this setup, diffraction will occur due to the finite size of the plane wave.
However, in this tutorial, we aim to focus solely on the physics of the *QWP* mirror, independent of the beam size. If possible, we would like to work under ideal conditions where the plane wave, mirror, and detector are all considered infinite in size.

In the software we can achieve this by using pointwise propagation throughout the optical setup, effectively eliminating diffraction effects.
(As a bonus, this also speeds up the simulations.)



Paraxial Approximation

The paraxial approximation sets *Ez* to zero and is valid from the source to the mirror and from the mirror to the detector. However, inside the coating, the plane wave's direction is not parallel to the mirror's axis, so the paraxial approximation should be disabled.
(Even though in the current simulation this has no effect on the results.)



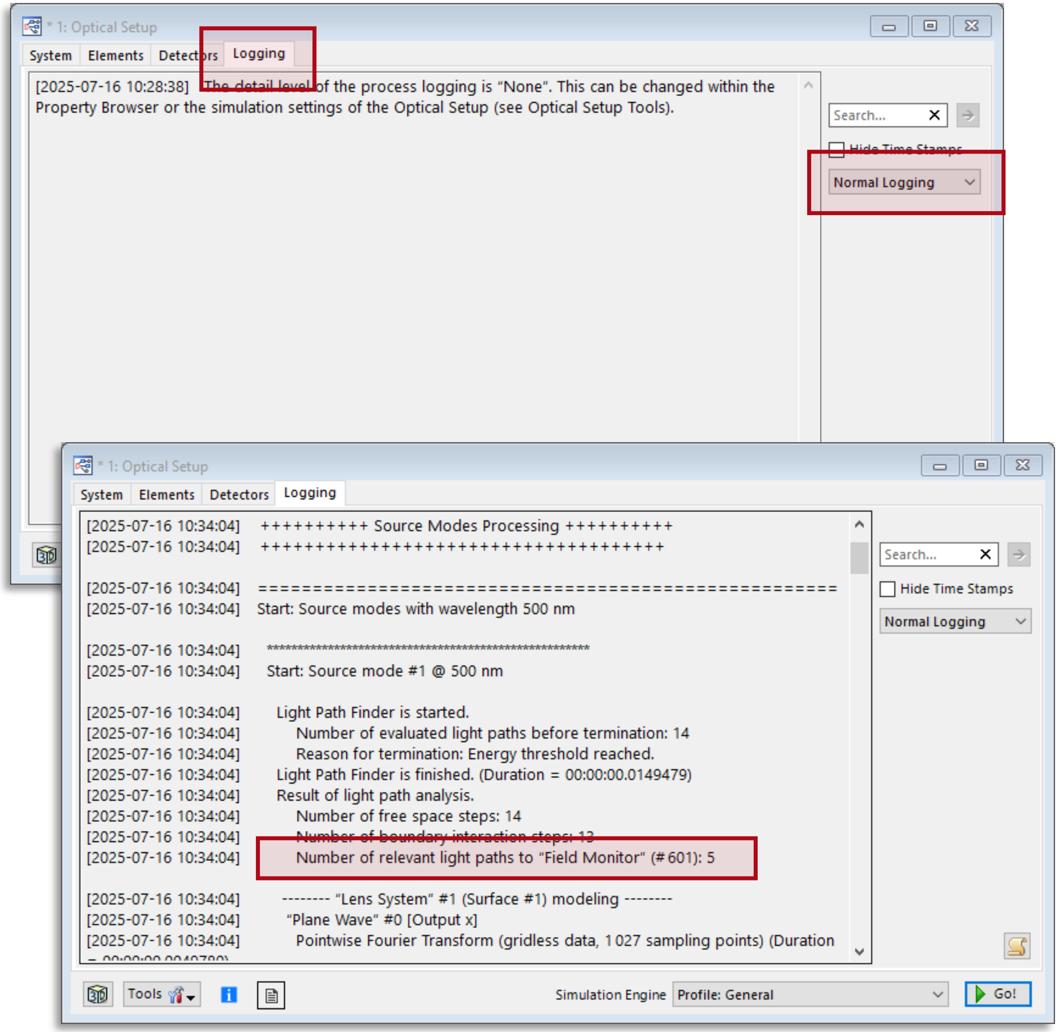
Evaluate Speed vs. Accuracy

In *Educational Tutorial: Fabry-Pérot Interferometer,* we already explored the concepts of convergence and precision.

While the coated mirror theoretically supports an infinite number of interfering paths, only a limited few contribute significantly to the result.

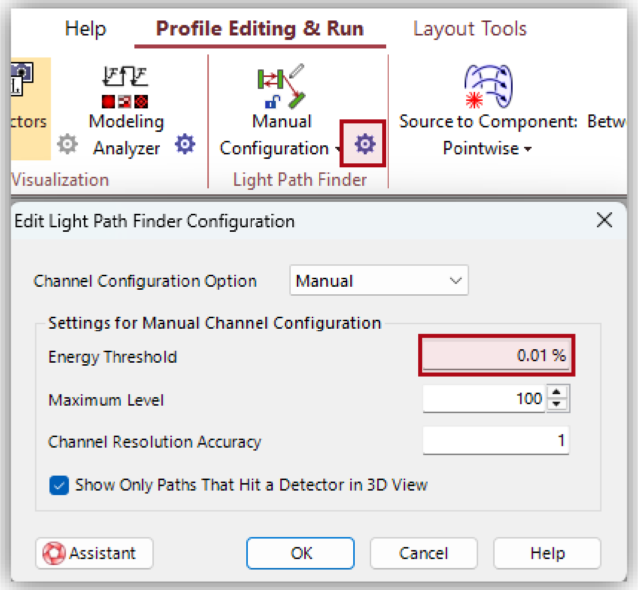
- Go to the *Logging* tab in the *Optical Setup Editor*
- select *Normal Logging*
- Click *Go!* to run the simulation
- Scroll to the top of the logging data
- Locate and review the following text

In *VLF*, simulations begin with the *Light Path Finder* routine, which runs partial simulations to identify possible light paths and estimate the energy they carry. Based on a predefined energy threshold, it has identified 5 relevant paths.



Evaluate Speed vs. Accuracy – Part 2

- Click the *cogwheel* icon in the *Light Path Finder*
- Note the default *Energy Threshold* (0.01%)
- Set it to 1e-04%, click OK
- Run the simulation and inspect the log output
- Repeat with *Energy Thresholds*: 1e-06%, 1e-08% and 1e-10%
- Observe that the number of paths increases
- Review the *Detector Results* (new entries appear at the top)
- Set *Energy Threshold* back to 0.01%



Based on the convergence, the default *Energy Threshold* of 0.01% appears sufficient to achieve ~ 0.5° accuracy for the current setup and will therefore be used moving forward.

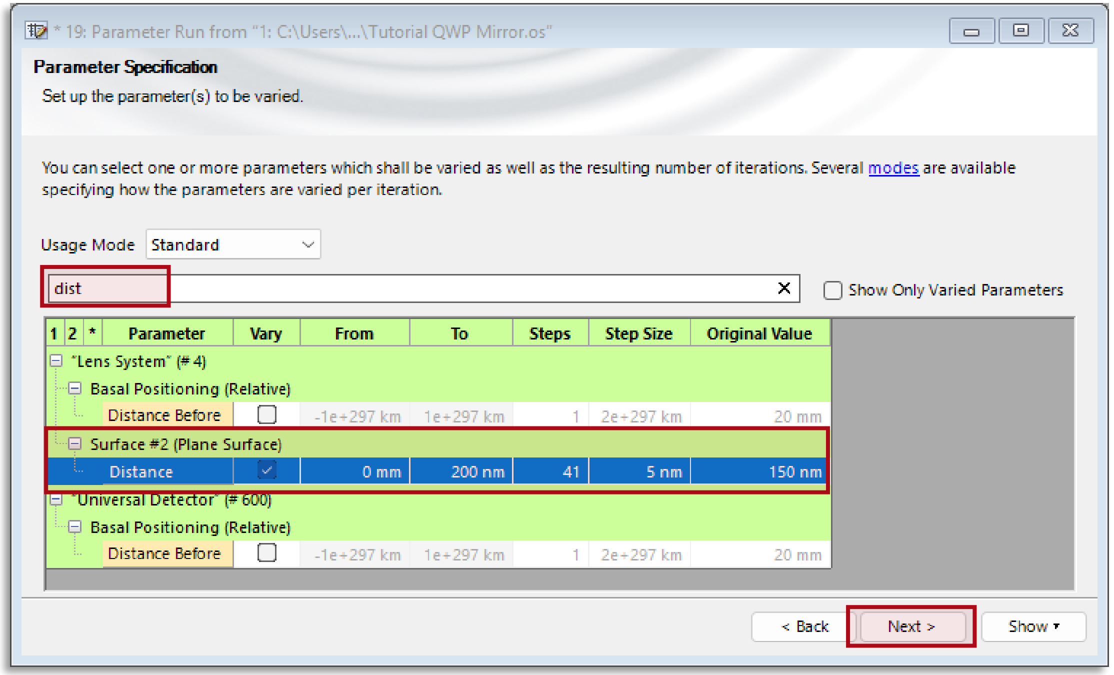
Detector Results

	Date/Time	Detector	Sub - Detector	Result	User Comments
5	2025-07-16 10:41:48	"Field Monitor" (# 601): Phae Offset Add-on (Profile: General)	Offset	-155.0928919°	
4	2025-07-16 10:41:07	"Field Monitor" (# 601): Phae Offset Add-on (Profile: General)	Offset	-155.0929095°	
3	2025-07-16 10:40:51	"Field Monitor" (# 601): Phae Offset Add-on (Profile: General)	Offset	-155.0905782°	
2	2025-07-16 10:40:27	"Field Monitor" (# 601): Phae Offset Add-on (Profile: General)	Offset	-155.1006471°	
1	2025-07-16 10:39:59	"Field Monitor" (# 601): Phae Offset Add-on (Profile: General)	Offset	-155.0777794°	

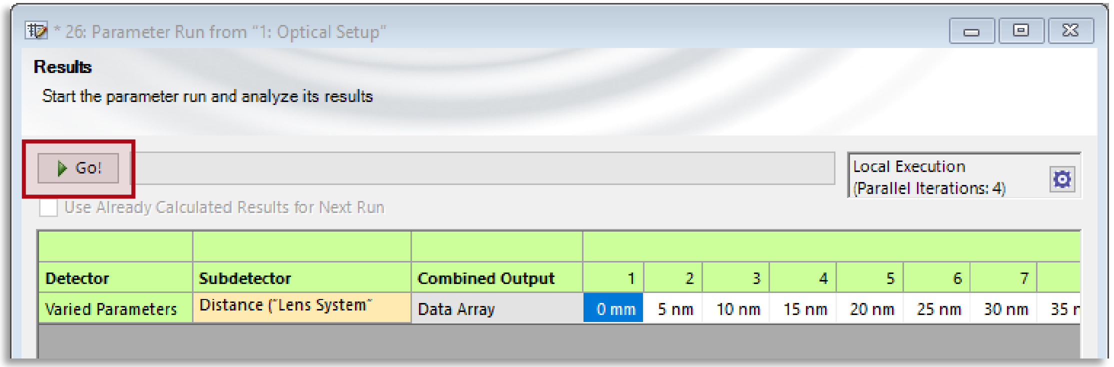
Attempt to Create a QWP Mirror at 45°

Let us vary the coating thickness to determine if we can create a *QWP* mirror at 45°

- Start a new *Parameter Run*
- Vary *Distance* of *Surface #2* from 0 to 200nm in steps of 5nm.

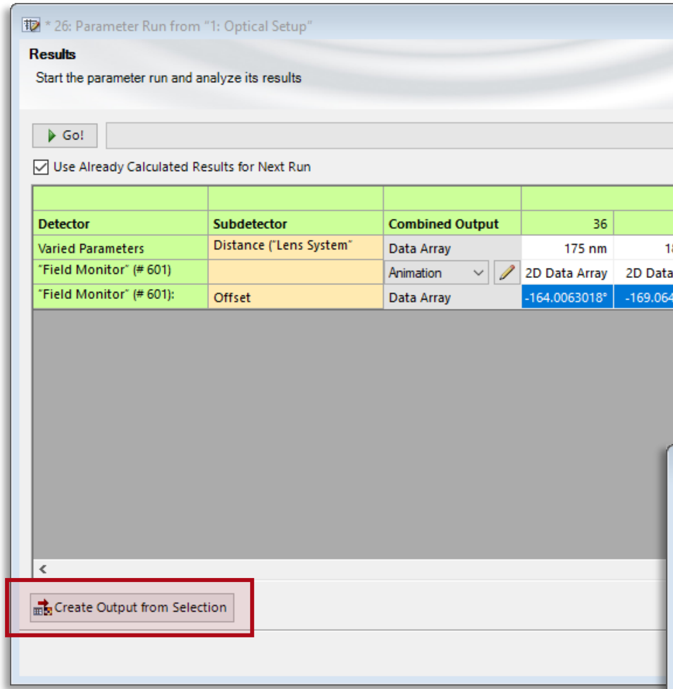


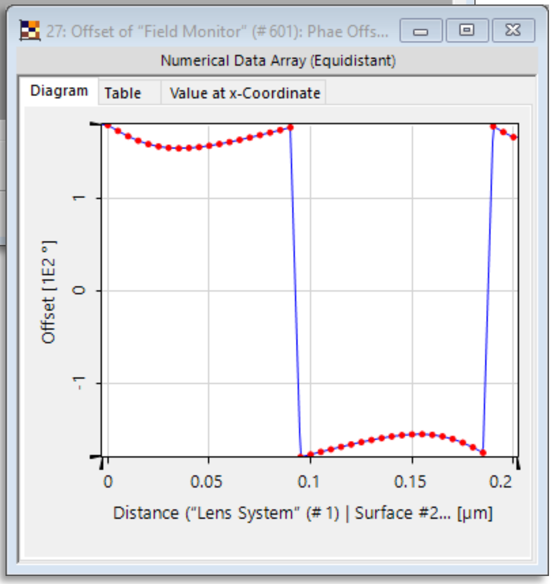
- Execute the *Parameter Run*



Attempt to Create a QWP Mirror at 45°

- Select the *Data Array* holding *Offset*
- Click *Create Output from Selection*





In the property browser:

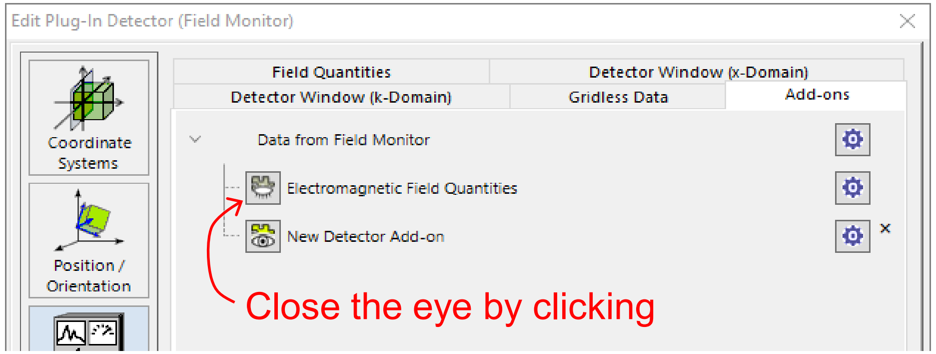
- set the format of the *Y-axis* to *Standard*
- In *Lines and Symbols* select
  - Dot

Conclusion
It appears that a quarter-wave plate (*QWP*) mirror at 45° cannot be realized with a single-layer coating, as the phase offset never reaches ±90° for any thickness. (Note that ±180° corresponds to a trivial reflection, simply producing the mirror image of the polarization.)

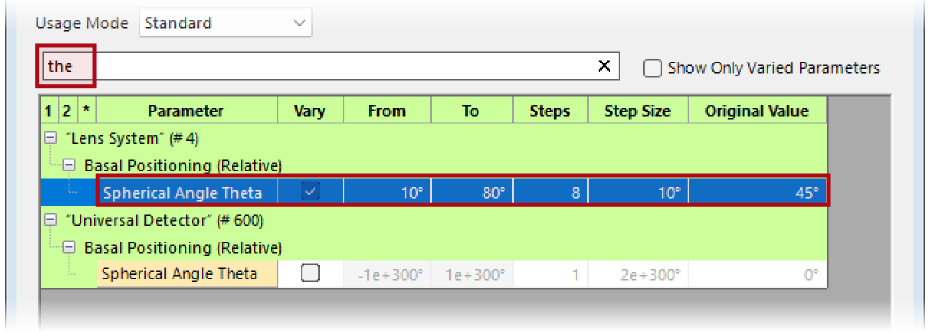
2D Parameter Run

Our goal is to explore the combinations of *Theta* and *Distance* that lead to circular polarization.

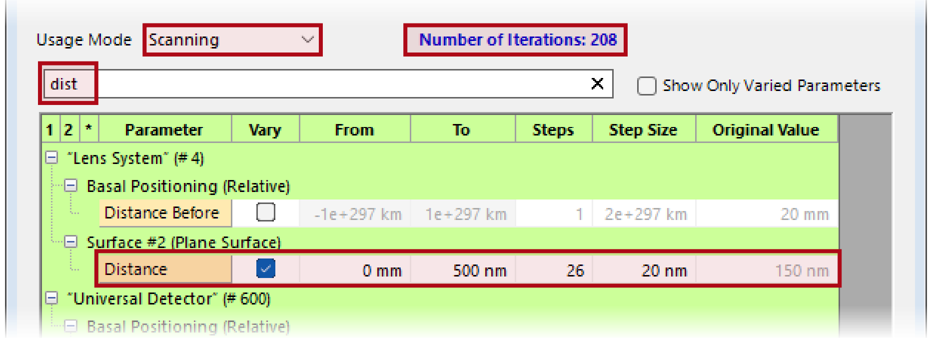
- Disable storing plots of the field detector to avoid excessive memory usage



- Open a new *Parameter Run* document
- Let *Theta* run from 10° to 80° in 8 *Steps*



- Let *Distance* run from 0 to 200 nm in 26 *Steps*
- Set *Usage Mode* to *Scanning* to cover all 8 × 26 combinations
- Note *Number of Iterations: 208*
- Execute the *Parameter Run*

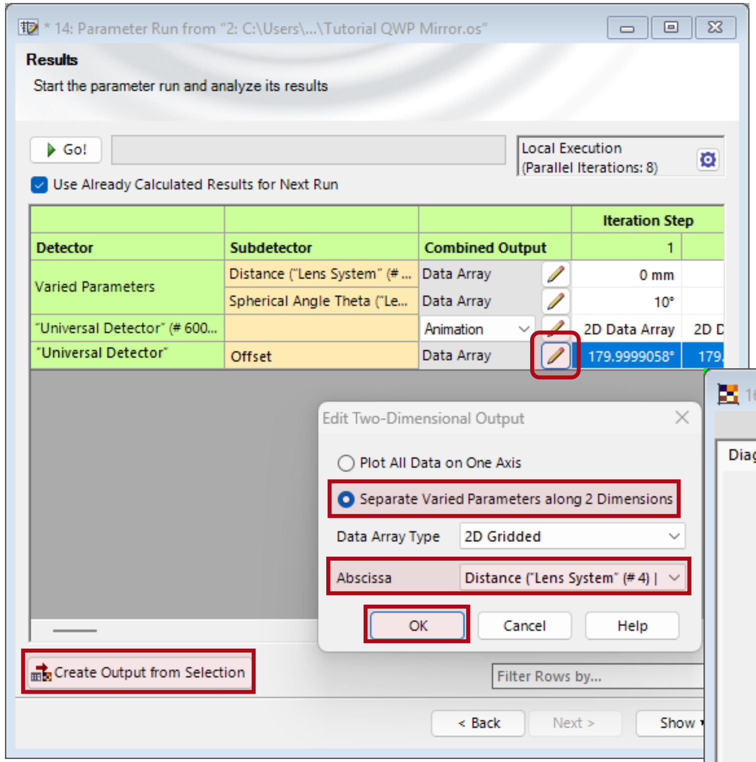


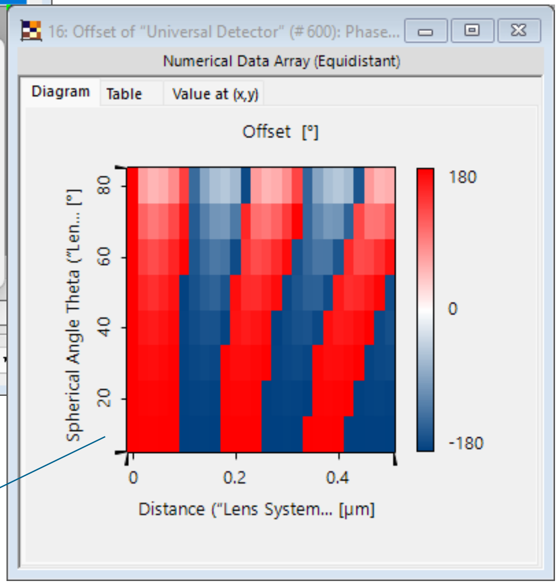
2D Parameter Run - Visualization

- Select the data array containing *Offset*
- Click the pencil icon
- Select *Separate Varied Parameters along 2 Dimensions*
- Set *Distance* as the *Abscissa*
- Click *OK*
- Click *Create Output from Selection*

In the *Property Browser*:

- *Set Format of Y-axis* to *Standard*
- Deselect *Auto Scaling of Data*
- Set *Displayed Data Range* to [-180°; 180°]
- Set *Format of Color Scale* to *Standard*





We want to clearly visualize the ±90° offsets. To do this, we will adjust the color table so that the plot changes color precisely at ±90°.

2D Parameter Run – Edit Color Table

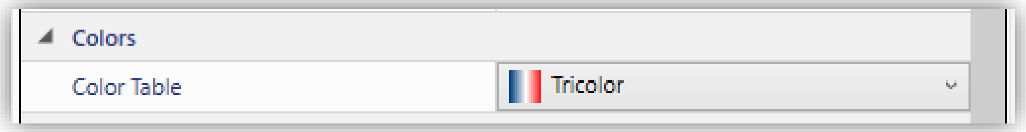
- In the *Property Browser*, ensure *the Color Table* is set to *Tricolor*.

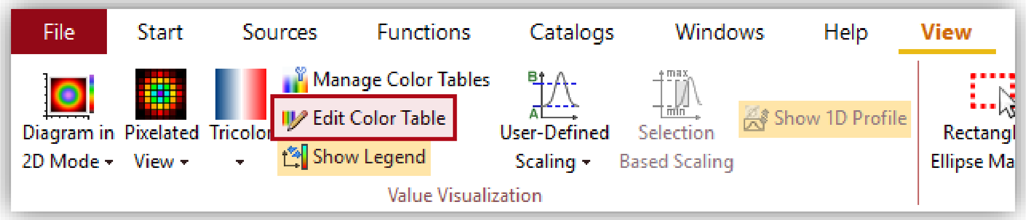
- In the *View Ribbon*, click *Edit Color Table*

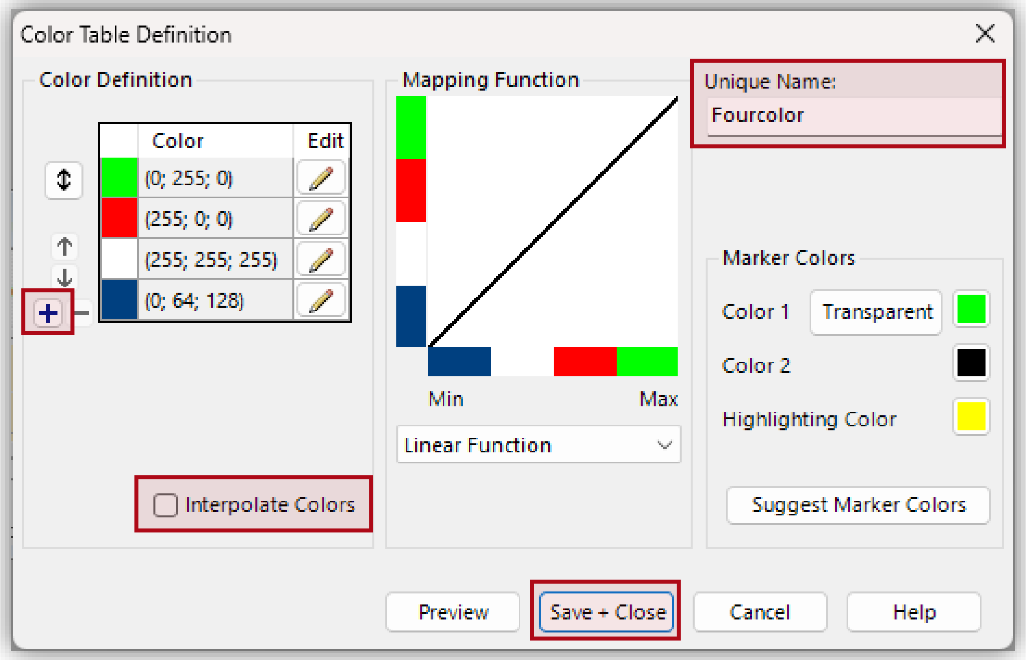
- Click + and add an additional color (e.g., green)
- Deselect *Interpolate Colors*
- Assign a *Unique Name* to the new color table
- Click *Save & Close*

The result:

- Values from 0 to 25% of the range will appear blue
- 25% to 50% will be white
- and so on

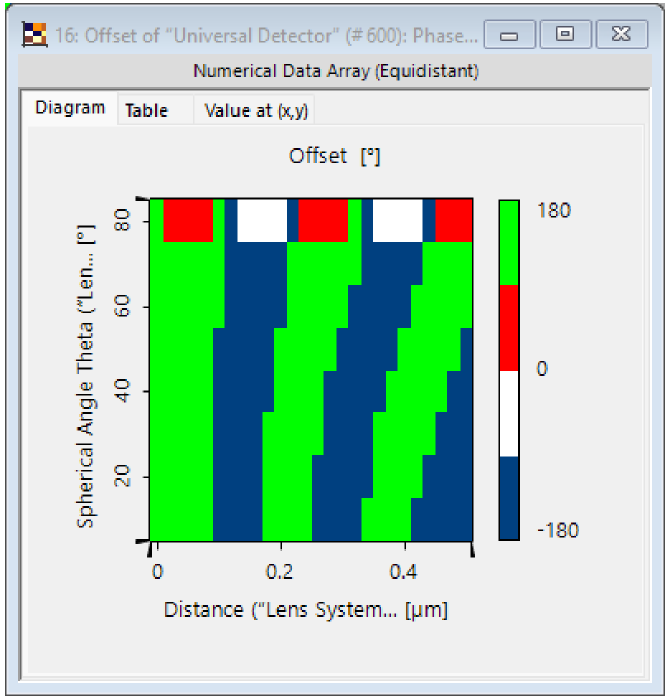






2D Parameter Run – Select Color Table

- In the *Property Browser*, select the newly created *Color Table*



At or near the red-green boundaries, the phase offset is expected to be +90°, while at or near the blue-white boundaries, it should be −90°.

The green-blue transitions indicate 2π phase jumps, which have no physical meaning due to the periodic nature of phase.

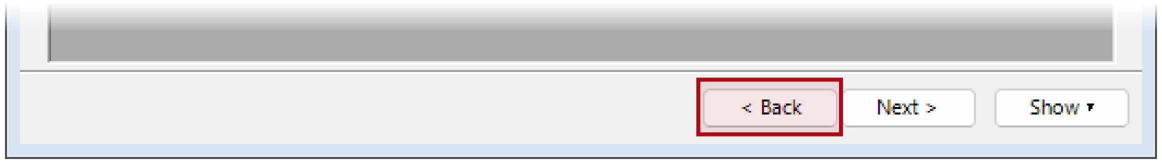
The plot suggests that solutions occur at relatively large angles of incidence. To investigate this further, we will increase the resolution by refining the step sizes for *Theta* and *Distance*, while limiting the *Distance* range to 0–200 nm and the *Theta* range to 60–80°.

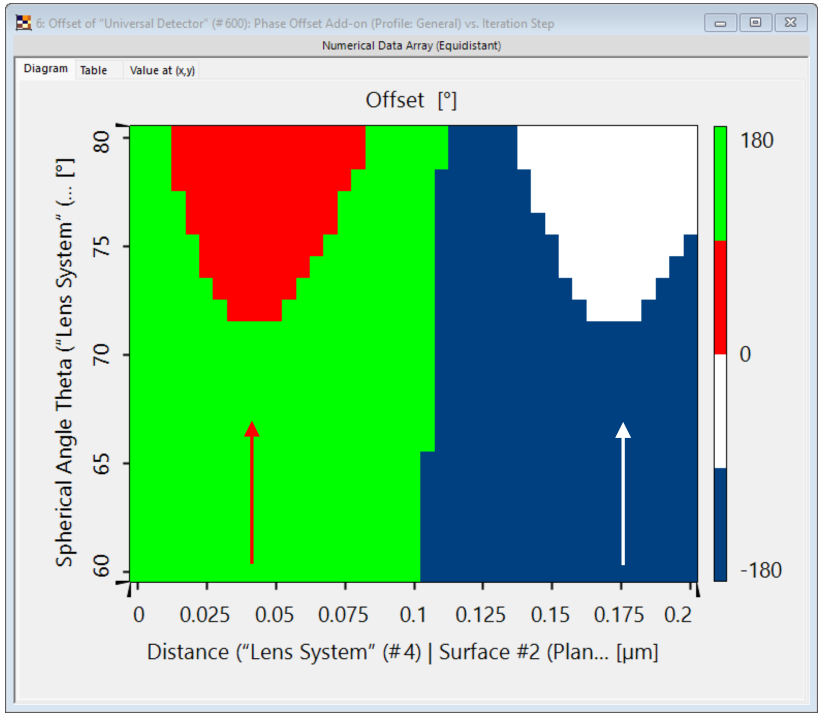
2D Parameter Run – Refine

- At the bottom of the *Parameter Run* document, click *Back* a few times, to return to the *Parameter Specification* window
- Vary *Theta* from 60° to 80° with Step Size 1°
- Vary *Distance* from 0 to 200 nm in steps of 5 nm
- Execute the *Parameter Run*
- Plot the result and format as before

The smallest angles of incidence appear to occur at ~ 42.5 nm for a phase offset of +90° and at ~ 172.5 nm for a phase offset of -90°, as indicated by the arrows.

In the next step, we will use the *Parametric Optimization* tool to find *Theta* for both of these coating thicknesses.

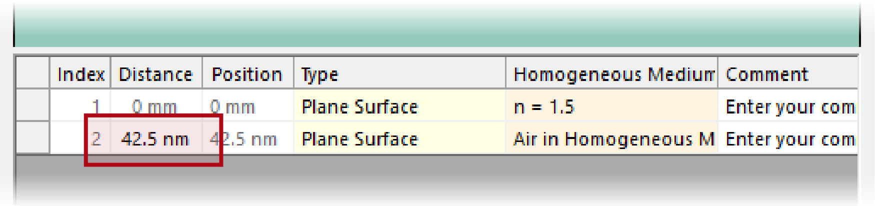




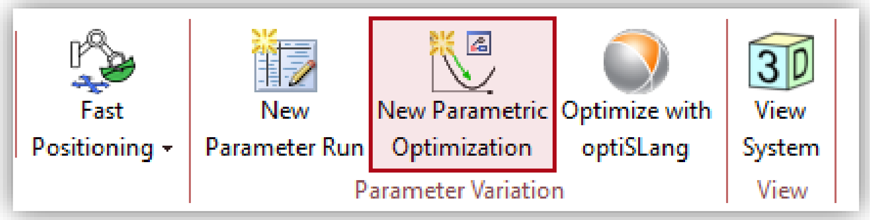
Tip: Use the *Point Marker* under *Selections* in the *Property Browser* to locate coordinates and read values on the plot.

Parametric Optimization

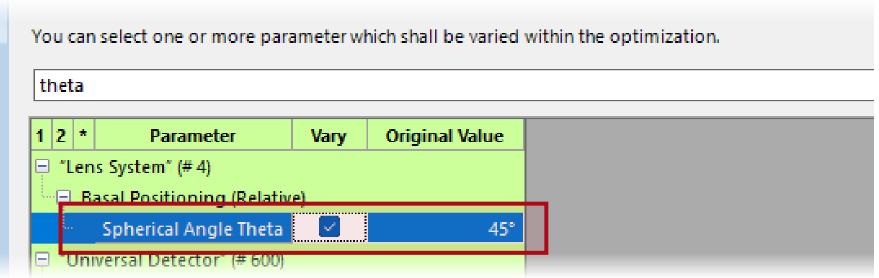
- Make the coating 42.5 nm thick



- Start a *Parametric Optimization*

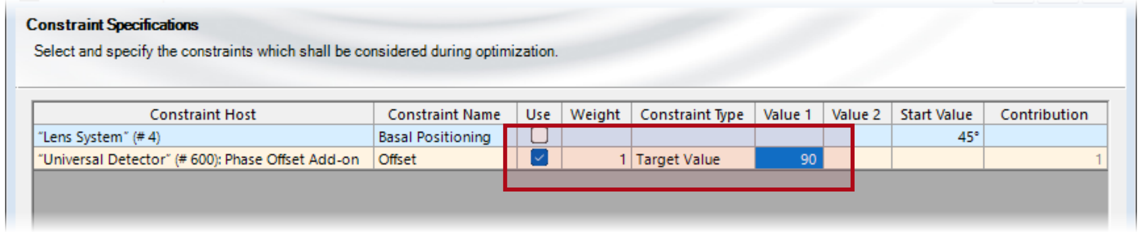


- Select *Theta* to be varied

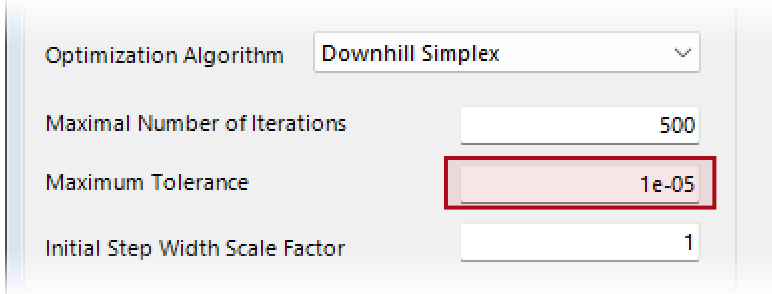


Parametric Optimization

- Enter the *Target Value* for *Offset* (90°)
- Click *Update*



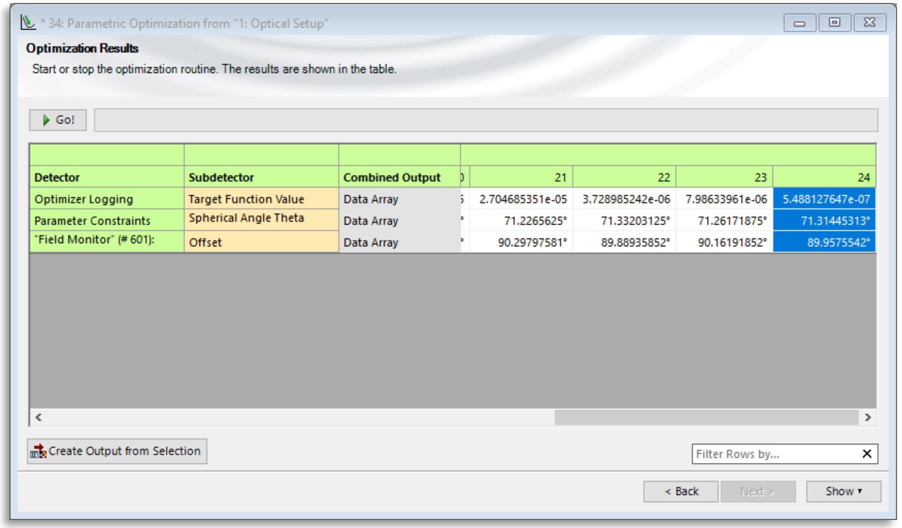
- Set *Maximum Tolerance* to 1e-05

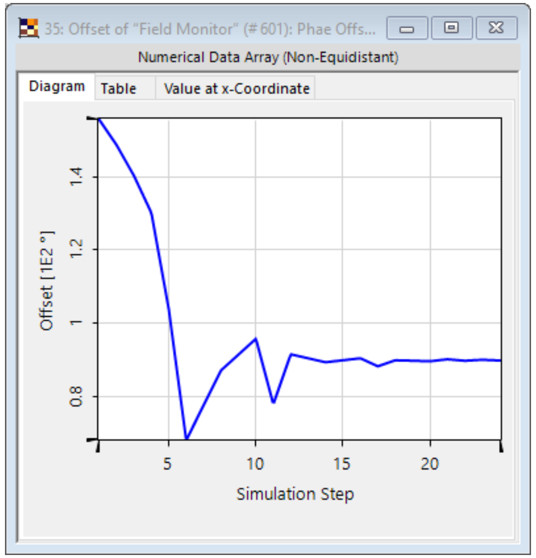


High accuracy will not be necessary, since we have limited the number of paths using an *Energy Threshold* of 0.01%.

Parametric Optimization – Part 2

- Run the optimization

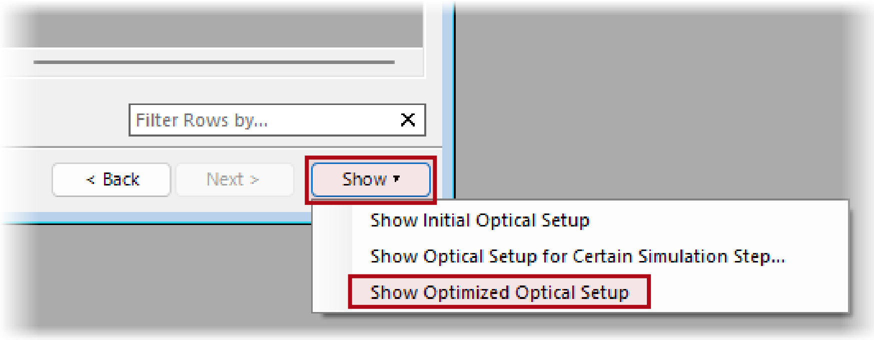




The routine used 24 steps for convergence

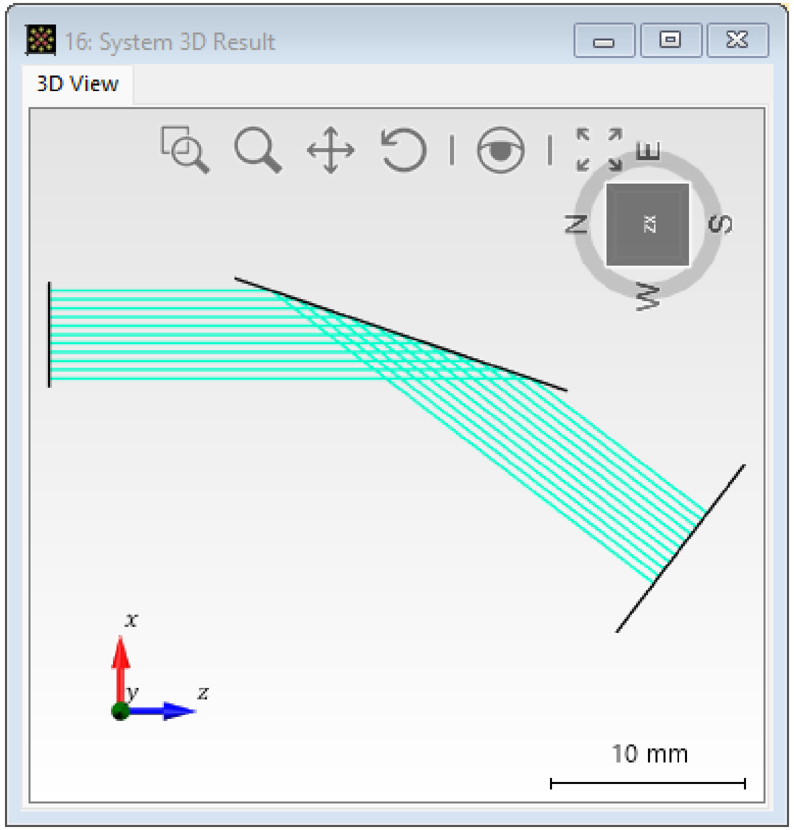
- Select *Show Optimized Optical Setup*

A copy of the *Optical Setup* will appear, with the only difference being that *Theta* now has the optimized value.

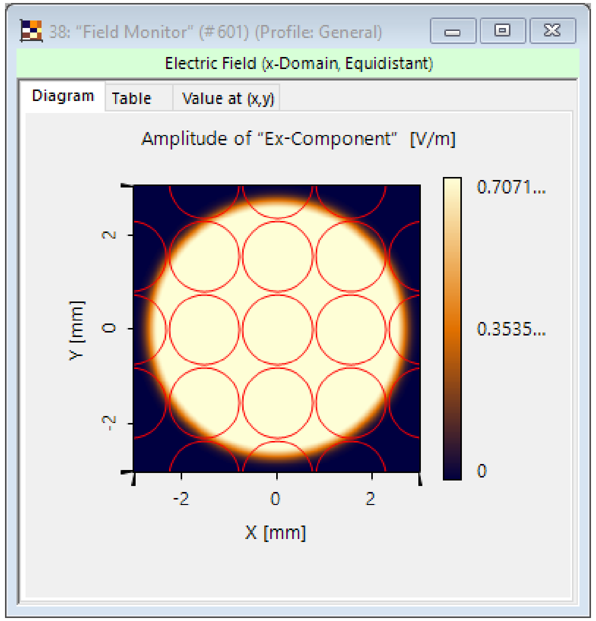


Optimized Optical Setup

- In the optimized setup, generate a 3D plot in the *Ray Results Profile*



- Enable the output of the *Electromagnetic Field Quantities* (open the eye)
- Run the simulation in the General Profile



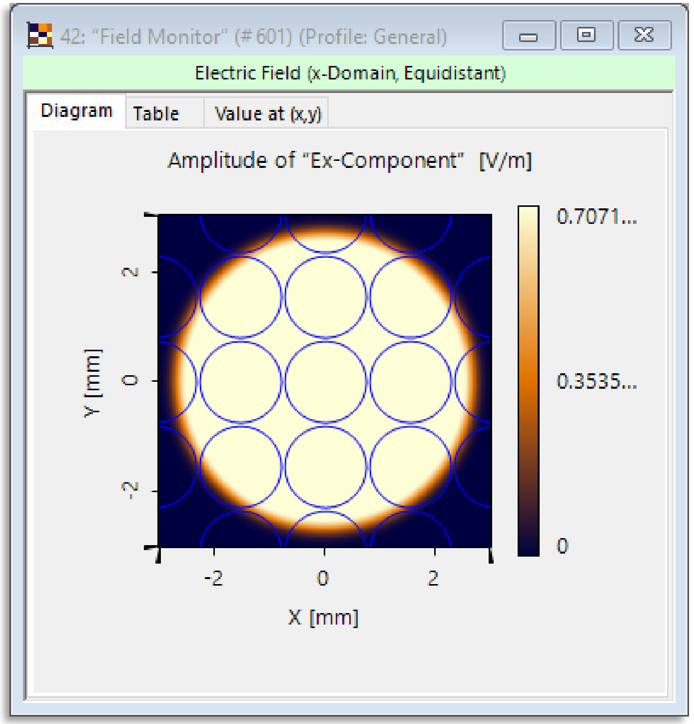
Indeed, the polarization is circular. The optimized setup and the optimization document can now be closed without saving.

Parametric Optimization – Different Thickness

- Change the coating thickness to 172.5 nm
- Run a new *Parametric Optimization*, this time with an *Offset Target Value* set at -90°

Offset	☑	1	Target Value	-90

Simulation Step		
22	23	24
3851404e-06	1.094502828e-05	4.560954968e-08
1.33203125°	71.26171875°	71.31445313°
9.92042585°	-90.18955318°	-89.98776369°



- Create the *Optimized Optical Setup*
- Open the eye
- Run the simulation in the *General Profile*

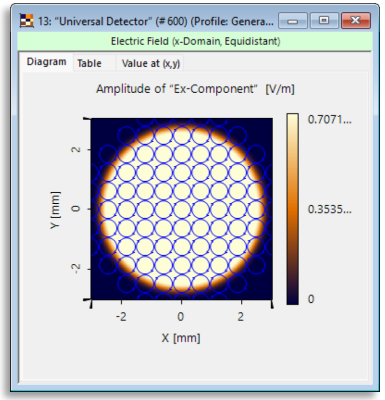
Again, the polarization is circular, but now the circles are blue.

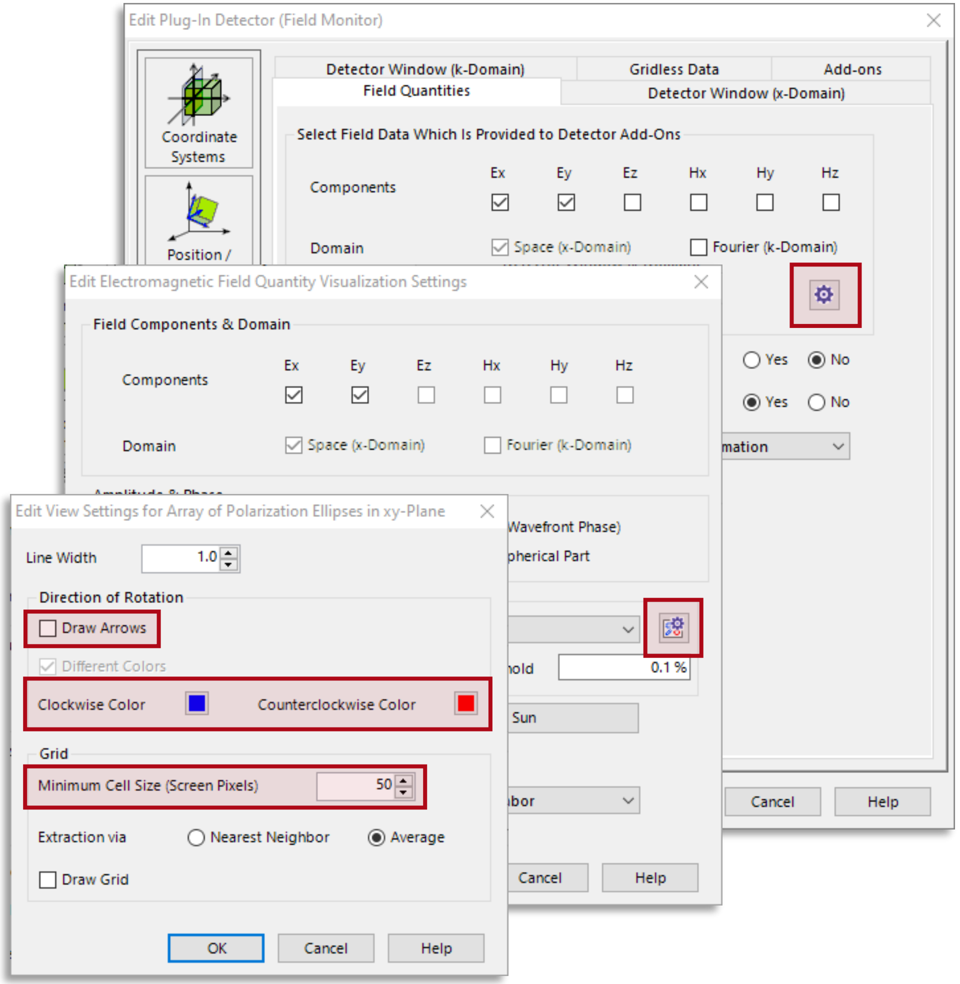
Polarization Ellipses Properties

- Continue working with the optimized setup
- Open the properties of the *Polarization Ellipses*

The color indicates the direction of rotation of the *E*-field as seen by the detector (Tip: The detector acts as the observer, looking towards the incoming light).

- Set *Minimum Cell Size (Screen Pixels)* to 20
- Select *Draw Arrows*
- Close all dialogs by clicking *OK*
- Run the simulation

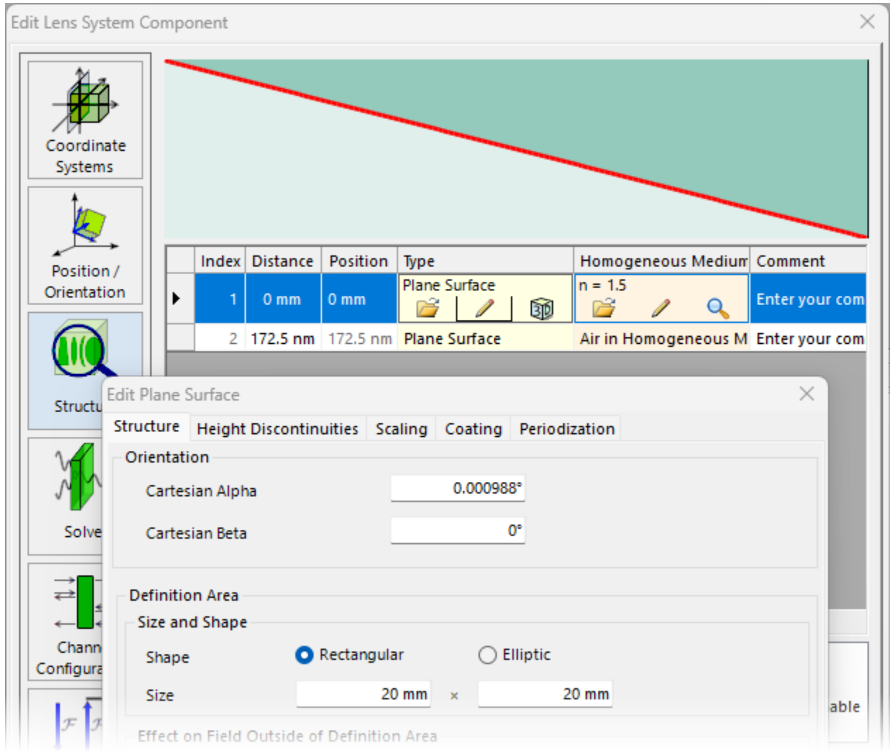




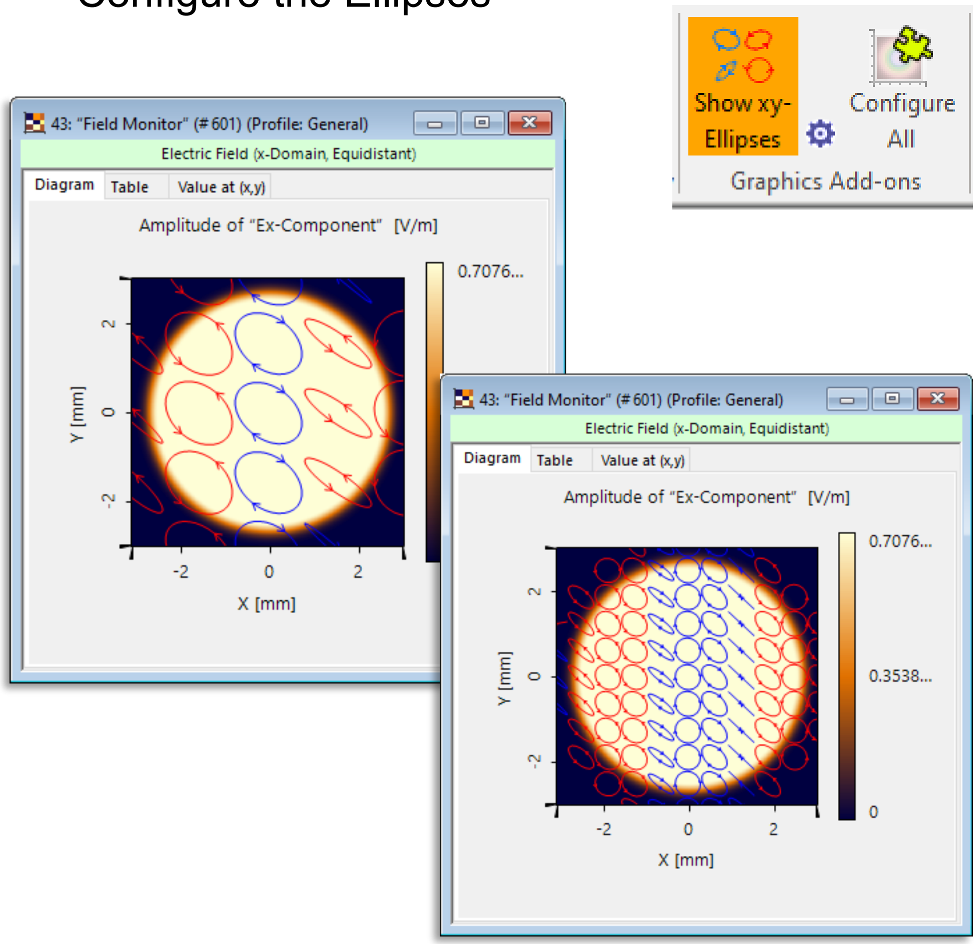
Ideas for Further Simulations – Wedge-Shaped Coating

- Set *Alpha* of the first surface to 0.000988°.

  This creates a wedge-shaped coating, with zero thickness at one end and a thickness of 172.5 nm at the center.

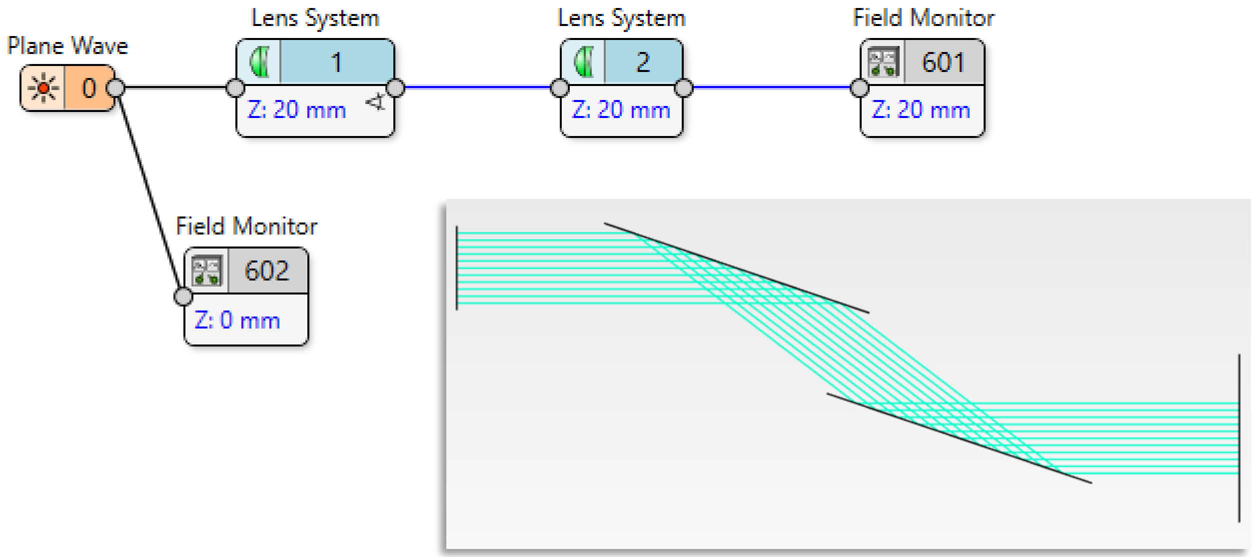


- Run the simulation
- Configure the Ellipses

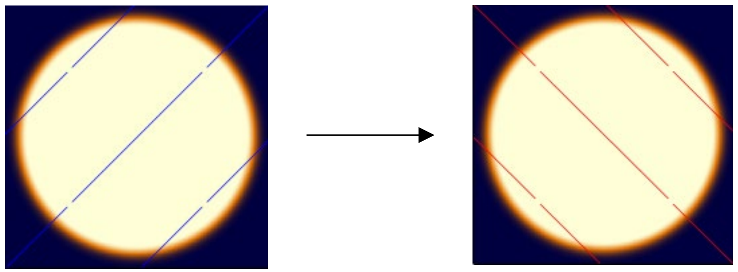


Ideas for Further Simulations – Two QWP Mirrors

- Set *Alpha* of the first surface back to 0°
- Set the *Energy Threshold* to 1e-04%
- Copy the *Lens System* and edit the setup to create the following configuration

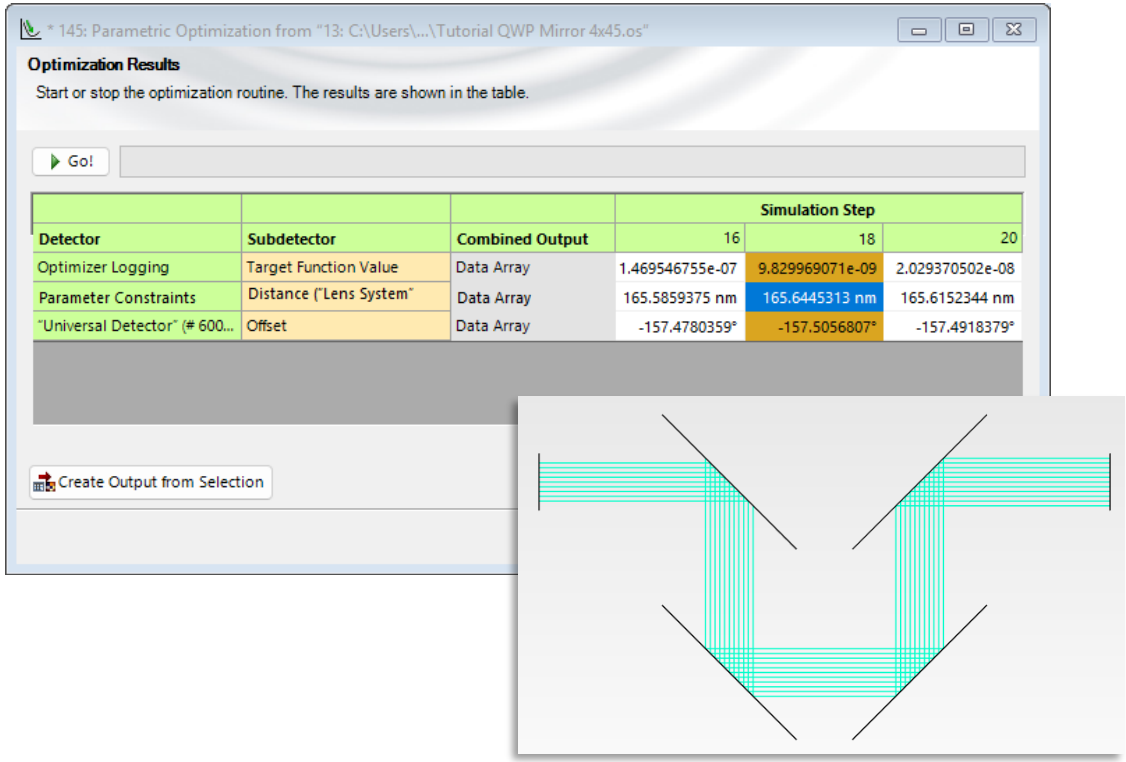


- Use coating thicknesses of 172.5 nm each
- Verify that the two mirrors together now function as a half-wave plate



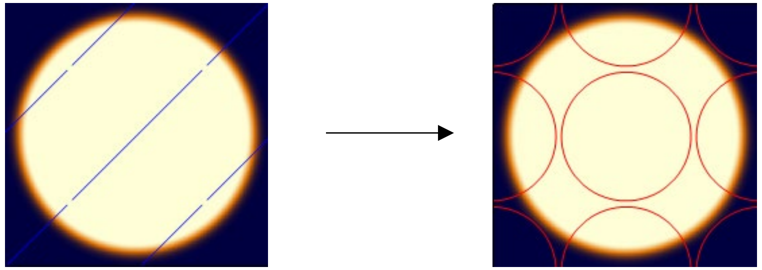
Ideas for Further Simulations – Four $\lambda/16$ Mirrors

- Use one mirror at 45°
- Notice that a reflection phase offset of –157.5° corresponds to a 22.5° shift, exactly one-sixteenth of a full wave (360°/16)
- Keep *Energy Threshold* = 1e-04%
- Use *Parametric Optimization* to find a coating thickness of app. 139.5 nm or 165.6 nm



- Continue with the *Optimized Optical Setup*
- Create a system of four $\lambda/16$ mirrors

- Set the *Energy Threshold* to 1e-06%
- Verify the generation of circular polarization



Conclusion

This second educational tutorial built upon the foundational skills introduced in the Fabry–Perot interferometer tutorial and guided you through more advanced features and applications of *VirtualLab Fusion (VLF)*.

You learned how to:

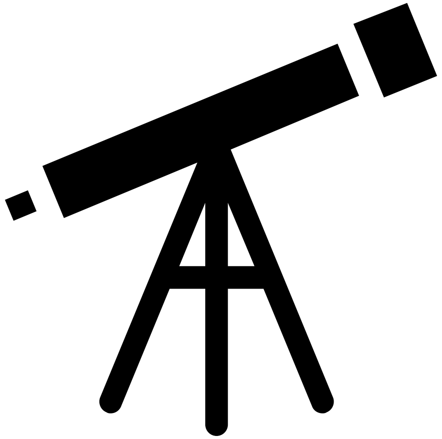
- Use *the Lens System* component to configure more complex optical setups,
- Program and apply a *Field Monitor Add-on* to extract customized simulation data,
- Conduct a 2D *Parameter Run* to explore the impact of multiple variables,
- Utilize *VLF's Optimization* tool to fine-tune system parameters, and
- Design a quarter-wave plate (*QWP*) mirror that converts linear to circular polarization—without the use of birefringent materials.

Conclusion

In addition to gaining deeper proficiency with *VLF's* tools, you explored how simple thin-film-coated mirrors can replicate the function of waveplates under the right conditions.

This hands-on example demonstrates how *VLF* can be used not only for simulating optical systems, but also for visualizing and understanding complex physical phenomena. If you encountered challenges along the way, we encourage you to revisit the first tutorial to reinforce the basics.

You're now well-prepared to continue exploring more advanced simulations and custom designs in *VLF*.



Document Information

title	Educational Tutorial: Quarter-Wave Plate Mirror
document code	TUT.0447
document version	1.0
required packages	-
software version	2025.1 (Build 1.176)
category	Tutorial
further reading	• Educational Tutorial: Fabry-Pérot Interferometer • Ellipsometry Analyzer