

LabVIEW 2012

New Feature Demo Script

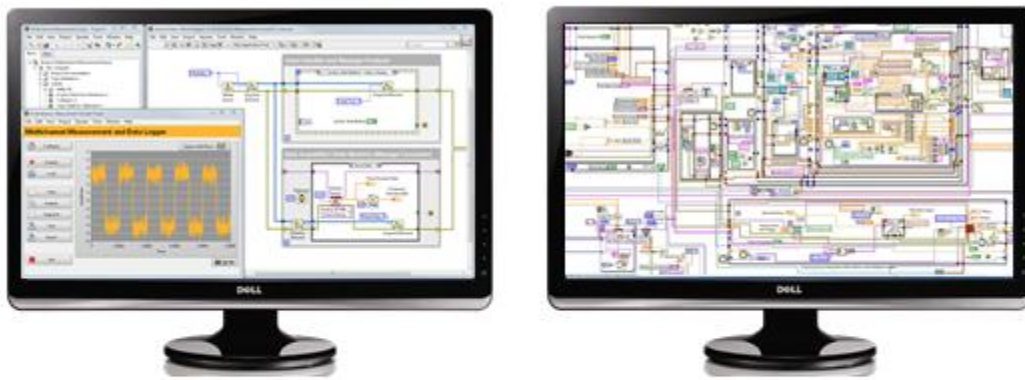
TABLE OF CONTENTS

Overview of LabVIEW 2012	3
Welcome Dialog and LabVIEW Skills Guide	4
Introduction to Templates and Sample Projects	7
Getting Started Window Improvements	12
Productivity Enhancements from the Idea Exchange	14
3D Stereo Vision	21
Sparse Matrix Performance	24
More Information	26

OVERVIEW OF LABVIEW 2012

Every new version of LabVIEW accelerates the productivity of engineers and scientists building any measurement or control application. Building a system fast is important, but it's equally important to build it *right*. Building a system right ensures quality and reliability, minimizes maintenance costs, enables team-based development, and guarantees the accuracy and performance of the system.

With this in mind, we created the new features and resources in LabVIEW 2012 to help you write better quality code and build your systems using good architectures and development practices, so that you can innovate with confidence and accelerate the overall success of your system. This technical manual will provide a brief overview of some of the new features in LabVIEW 2012. For more detailed information, visit ni.com/labview/whatsnew



Build This. Not That.
LabVIEW 2012 helps you write better code

WELCOME DIALOG AND LABVIEW SKILLS GUIDE

GOAL

Familiarize existing users with the changes and improvements we've introduced to the LabVIEW 2012 getting started window.

SCENARIO

This is what every user will see when they launch LabVIEW 2012.

CONCEPTS COVERED

- Welcome dialog
- LabVIEW Skills Guide

SETUP

To completely reset this demo, you'll want to have the Getting Started Window appear exactly as a user would see it the first time they launch. To do this, you'll want to remove the list of created projects and existing projects and reset other default settings. The simplest way to do this is by temporarily replacing your 'labview.ini' file. If you're not sure how to do this, please follow these steps:

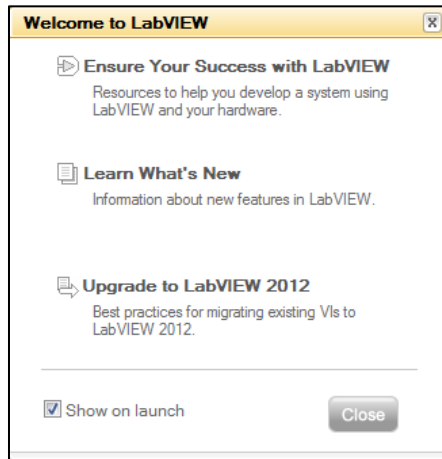
1. Close LabVIEW 2012 if it's already open
2. Navigate to 'C:\Program Files\National Instruments\LabVIEW 2012'
3. Find the 'LabVIEW.ini' file in this directory
4. Rename it ('LabVIEW_backup.ini' recommended)
5. Launch LabVIEW
6. The Getting Started Window should appear with default settings

You can restore your environment settings after this demo by following these steps

7. Close LabVIEW 2012
8. Navigate to 'C:\Program Files\National Instruments\LabVIEW 2012'
9. Delete the 'LabVIEW.ini' file that was created
10. Rename 'LabVIEW_backup.ini' to 'LabVIEW.ini'
11. Restart LabVIEW

Step 1: Welcome Dialog Explanation

1. When LabVIEW launches for the first time you'll see a new 'Welcome to LabVIEW' dialog. The resources this points to are designed to help you use the product successfully by pointing you to resources and learning materials on the environment, new features, and how to upgrade existing applications.




2. Detailed information about all the new features in the platform can be found by clicking 'Learn What's New,' which will launch the 'LabVIEW 2012 Features and Changes' help document. Additional resources on new features can also be found online at ni.com/labview/whatsnew
3. Best practices for upgrading existing LabVIEW systems is available at the third link, which provides a set of step-by-step instructions for how to mitigate the risk associated with major upgrades. It is recommended that you review this before saving previous projects in LabVIEW 2012.
4. Click 'Ensure Your Success with LabVIEW' to launch the LabVIEW Skills Guide.

Step 2: Explore the LabVIEW Skills Guide - *Required Internet*


1. The LabVIEW Skills Guide is a comprehensive resource to find online tutorials and training that will help you develop any LabVIEW-based system. The skills guide enumerates all of the recommended best-practices and technologies based on the complexity of your application and the hardware that it includes. New users can use the skills guide to find online resources that will help them learn the environment, while advanced users can explore new concepts or revisit topics in order to brush-up on their skills.
2. Identify the relevant software concepts by first determining which of the listed descriptions best suits you and your application.
3. Click on the appropriate title to see a list of recommended concepts and technologies that your application should use.
4. The skills guide will provide links to some or all of the following for software concepts
 - a. Free online resources and web tutorials
 - b. Self-Paced Online training (this will be explored in greater detail in Step 3)
 - c. Instructor-led Training

<p>Software Engineer</p> <p>Design and develop a medium to large application (20 to 100 VIs) Develop one or more systems for ongoing use or deployment over multiple months or years Develop any applications to be used, supported, or maintained by others Plan to use LabVIEW for multiple projects Use LabVIEW regularly</p>			
<p>▾ Sound like you?</p>			
Required Skills	Online Product Documentation Free written documentation available 24/7 on ni.com	Self-Paced Online Training Online video training available 24/7; receive with active software service contract or purchase individually	Instructor-Led Training Live classes taught online or in a classroom by certified instructors; provides exercises and hands-on hardware experience; available for purchase
Install LabVIEW	LabVIEW Installation Guide		
Navigate the LabVIEW environment	Getting Started with LabVIEW	LabVIEW Core 1	LabVIEW Core 1
Apply key LabVIEW structures (such as Case structures and loops) and data types (such as Booleans, strings, and numerics)			
Apply key LabVIEW elements for relating data (such as arrays, clusters, and typedefs)			
Read and interpret existing LabVIEW code			
Troubleshoot and debug LabVIEW code			
Understand and select appropriate application timing techniques	Controlling Timing in LabVIEW Applications		
Apply basic design patterns and LabVIEW templates such as the Simple State Machine template	Simple State Machine Template		
Use event programming to handle user interface interaction or communicate data between processes	Event-Driven Programming		


- Click on the second tab at the top 'Step 2: Hardware Skills'
- Identify the relevant hardware platform or application area to explore. All of them also present a spectrum of resources for users to browse based upon the hardware requirements and system complexity.




Data Acquisition
M Series, X Series, CompactDAQ, and Stand-Alone DAQ



Instrument Control
GPIB, Serial, USB, and Ethernet



Embedded Control and Monitoring
CompactRIO and NI Single-Board RIO



Automated Test
PXI and Modular Instruments Hardware, NI TestStand and IIL VeriStand Software

Basic Performance (Scan Engine) Prototype Build a functional prototype or short-term use system
 Sample or update all I/O channels at $\leq 500\text{ Hz}$ and use software-based control or safety logic

▾ Sound like you?

Basic Performance (Scan Engine) Deployment Develop one or more systems to be used on an ongoing basis over multiple months or years
 Sample or update all I/O channels at $\leq 500\text{ Hz}$ and use software-based control or safety logic

▾ Sound like you?

Required Skills	Online Product Documentation Free written documentation available 24/7 on ni.com	Instructor-Led Training Live classes taught online or in a classroom by certified instructors; provides exercises and hands-on hardware experience; available for purchase
Setup	Install and configure CompactRIO hardware and LabVIEW software	Getting Started
Implement Windows Host Application	Design a host application that can handle user events and display data Implement network communication between processes	LabVIEW Core 2

- Return to the software tab and expand one of the options that list three columns of links. The middle column is for self-paced online training, which we will explore next.

INTRODUCTION TO TEMPLATES AND SAMPLE PROJECTS

GOAL

Explore how to create and use templates and sample projects when starting a new application.

SCENARIO

Clicking 'Create Project' will prompt a user to select a template or sample project to be a building block for a new project.

CONCEPTS COVERED

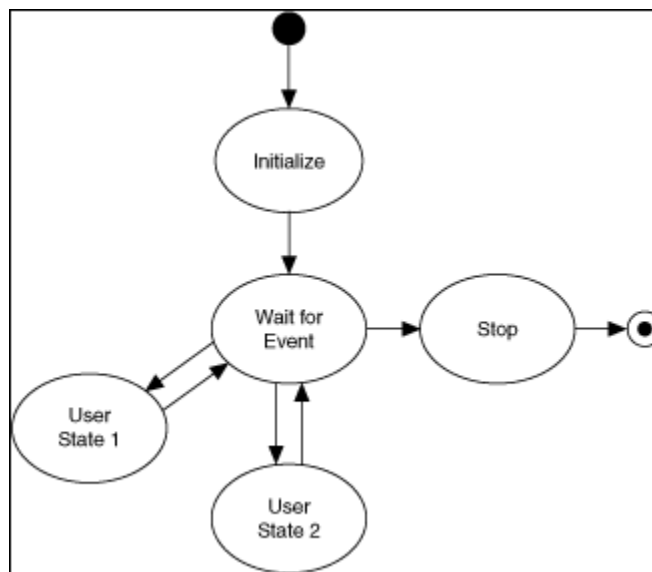
- Template and Sample Project documentation
- Custom icon overlays
- 'Code Recommended' and 'Code Required' comments
- Using a Sample Project with Hardware

SETUP

- The list of templates and sample projects will depend on what components of the platform you have installed.
- To install the sample projects for use with DAQ hardware, please be sure to install NI DAQmx 9.5.5 or later
- Simulated hardware can be used with this exercise, but it is recommended that you have an analog input capable of reading a voltage value available.

Step 1: Browse the List of Available Templates and Sample Projects and view Documentation

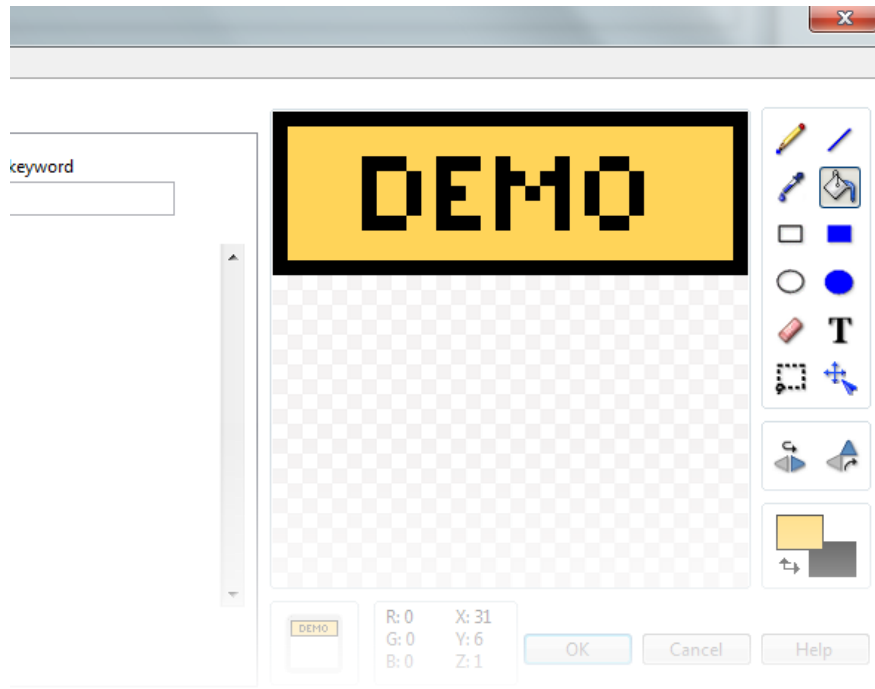
1. Click 'Create Project' in the Getting Started Window. This launches the 'Create Project Dialog' which lists available templates and sample projects that you can use as a starting point for a new application.
2. The italicized text next to every item indicates whether it is a template or a sample project. Templates illustrate the fundamental LabVIEW design patterns, and sample projects illustrate the combination of one or more of those design patterns for a specific application.
3. Highlight the 'Simple State Machine' and click 'More Information' to open the documentation. Every template and sample project includes an HTML document that explains the fundamental design and the concepts that a user will need to be familiar with. The templates have especially detailed documentation, as these are the building blocks for the more complex sample projects.
4. The documentation describes use cases for this particular template. The last bullet under 'Use Cases' in the documentation describes that this template should be used for '*An application that takes one measurement, logs it to disk, and then waits for another user action. The states in this application might include waiting for user input, performing the measurement, logging the data, displaying the data, and so on.*'
5. Scroll down to the 'Overview' section to display the flow chart for this diagram. This shows the logical relationship between the cases in the case structure.



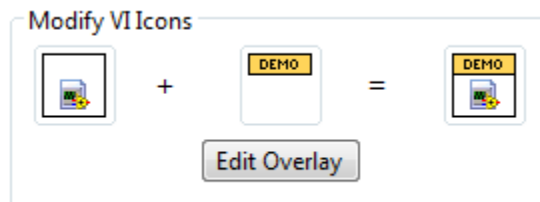
6. Return to the 'Create Project' dialog (which should be behind the window displaying the documentation).

Step 3: Change Icon Overlay and Open a Template

1. Select 'Simple State Machine' and click 'Next.' All of the template and sample projects have additional dialogs where you can customize a variety of things, including file names and icons.
2. Change the Project Name to be 'Simple Measurement Application'
3. Click 'Edit Overlay' to set a common overlay that appears on all the VIs that will be generated.
4. Use the paint and text tools to reproduce the following change to the overlay:



5. Click 'OK.' Your modifications should now be shown in the Create Project dialog.

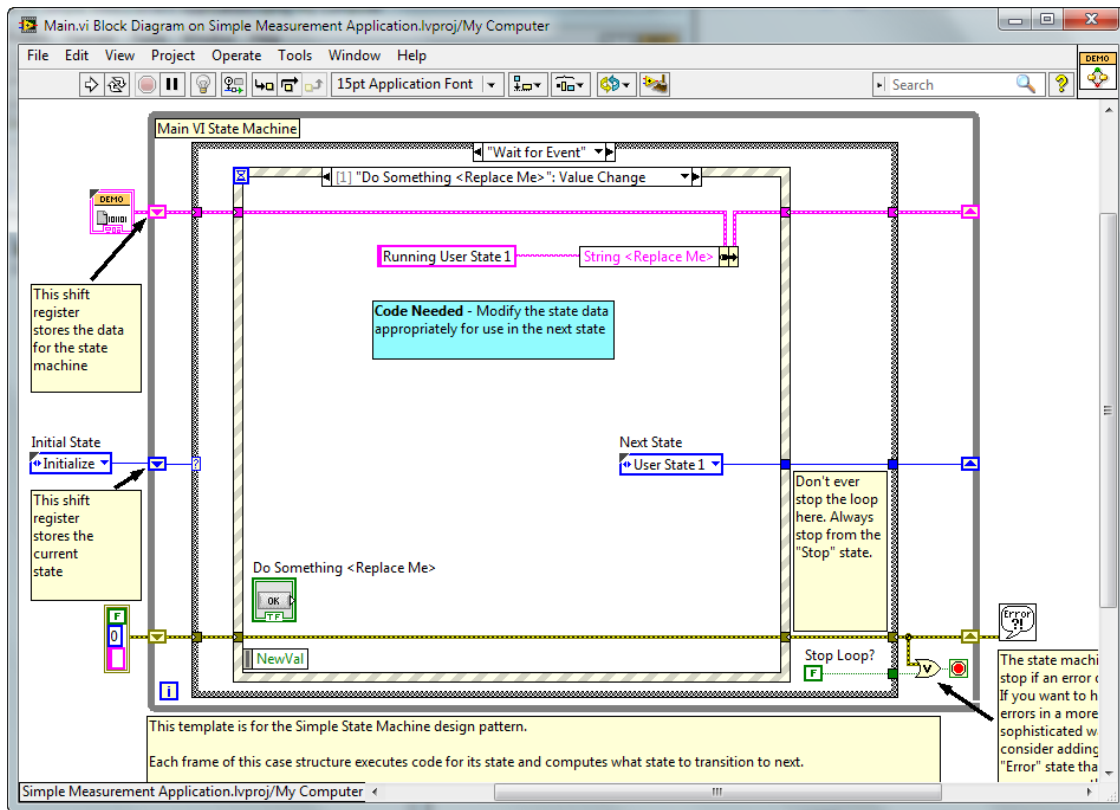


6. Click 'Finish' to begin creating a new copy of the template in a new Project Explorer window. Any modifications that are made to this copy will not override the original template.

Step 3: Review the Instructions on How to Customize and Change Code

1. Open main.vi and navigate to the block diagram (Press CTRL + E)
2. Templates are extensively documented to clearly indicate functionality and where code is either needed or recommended. Blue comments are used to clearly indicate where code is needed.
3. The 'Initialize' state is the first state that gets executed, as determined by the enumerated constant on the left side of the while loop. As the comments indicate, this is where you'll want to initialize the display and the data that is stored in the shift register. After this state, the application goes to the 'Wait for Event' state, where it stays until a user triggers an event.
4. Change the visible case to the 'Wait for Event' case. This case contains an Event Structure that captures user interaction with front panel control and selects the next state the state machine will enter. If, for example, the user presses 'Do Something,' the event structure will set the next state as 'User State 1' and store 'Running User State 1' as the data in the shift register.

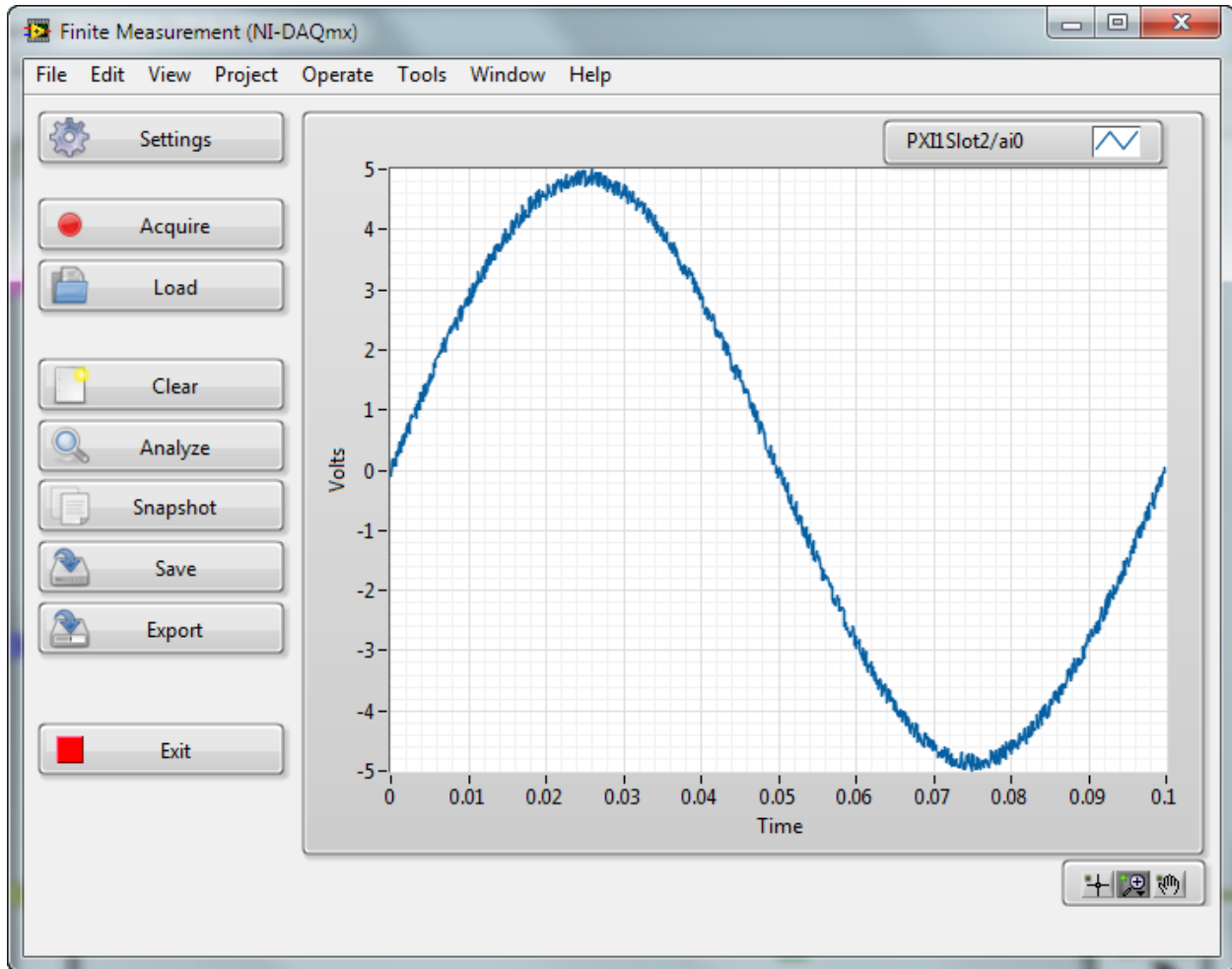
5. Select the 'User State 1' case in the case structure. All it does is display the contents of the shift register. All of the diagrams in these structures clearly indicate where code is needed or needs to be modified.



6. Close the Project Explorer to exit this project - next we will examine the use of this template for a complete, ready-to-use finite measurement application.

Step 4: Open a Sample Project and run with DAQmx Hardware

1. Click 'Create Project' from the Getting Started Window
2. Select the 'Finite Measurement (NI-DAQmx)' Sample Project and select 'Next.'
3. Give it a unique name on the next page and click 'Finish'
4. Open main.vi and note that it is a much more complete application, but it's based entirely off of the state machine template.
5. If it's not already, plug in a DAQ device that has an analog voltage input (such as a CompactDAQ). If plugging in hardware for the first time, ensure that Windows has recognized and installed the drivers for the hardware (this should take no longer than 30 seconds).
6. Once the hardware is installed, click 'run' on the front panel.
7. Click 'Settings,' which will display a list of all the physical channels on your system and other settings for the measurement. Select the appropriate channel and hit 'OK'
8. Click 'Acquire' to read the signal



9. Click 'Analyze' to display some basic characteristics, including the mean value.
10. Other functionality that this application has built into it includes:
 - a. Saving to an *.lvm file
 - b. Saving to an image
 - c. Loading a previously saved *.lvm file
11. Click 'Exit' to stop the application
12. View the block diagram (Press CTRL + E) to show that this sample project is based on the simple state machine template.
13. Close this project and return to the Getting Started Window.

GETTING STARTED WINDOW IMPROVEMENTS

GOAL

Familiarize users with major improvements in the Getting Started Window.

SCENARIO

The Getting Started Window will be displayed after hiding or closing the Welcome Dialog.

CONCEPTS COVERED

- Pinning commonly used projects
- RSS feed
- Tools network integration

SETUP

- Ensure VI Package Manager is installed

Step 1: Pin Projects in Getting Started Window

1. The Getting Started Window should now be populated with the names of the projects that we just opened. The items on the left are the templates and sample projects that we just used to create new projects. The list on the right is the specific projects we just created.
2. Hover over the items in the list and click the pin to 'pin' the item in this dialog. The new Getting Started Window allows us to pin items that are commonly used. This is especially useful if we're actively developing multiple projects, but want to ensure that key projects are always readily available at the top of the list.

Step 2: Getting Started Window RSS Feed

1. The bottom of the getting started window now features an RSS feed where you can see updates and news regarding LabVIEW. Click on this to catch up with 'LabVIEW in Action'

Step 3: LabVIEW Tools Network Integration

1. Click 'Find Drivers and Add-ons.' The bottom link, 'Find LabVIEW Add-ons' launches and in-product experience powered by JKI's VI Package Manager, which is now included on the platform DVD. The fact that it's included makes it even easier to create and share package files with other developers.

PRODUCTIVITY ENHANCEMENTS FROM THE IDEA EXCHANGE

GOAL

Explore some of the productivity enhancements that have been added based on feedback from the community

SCENARIO

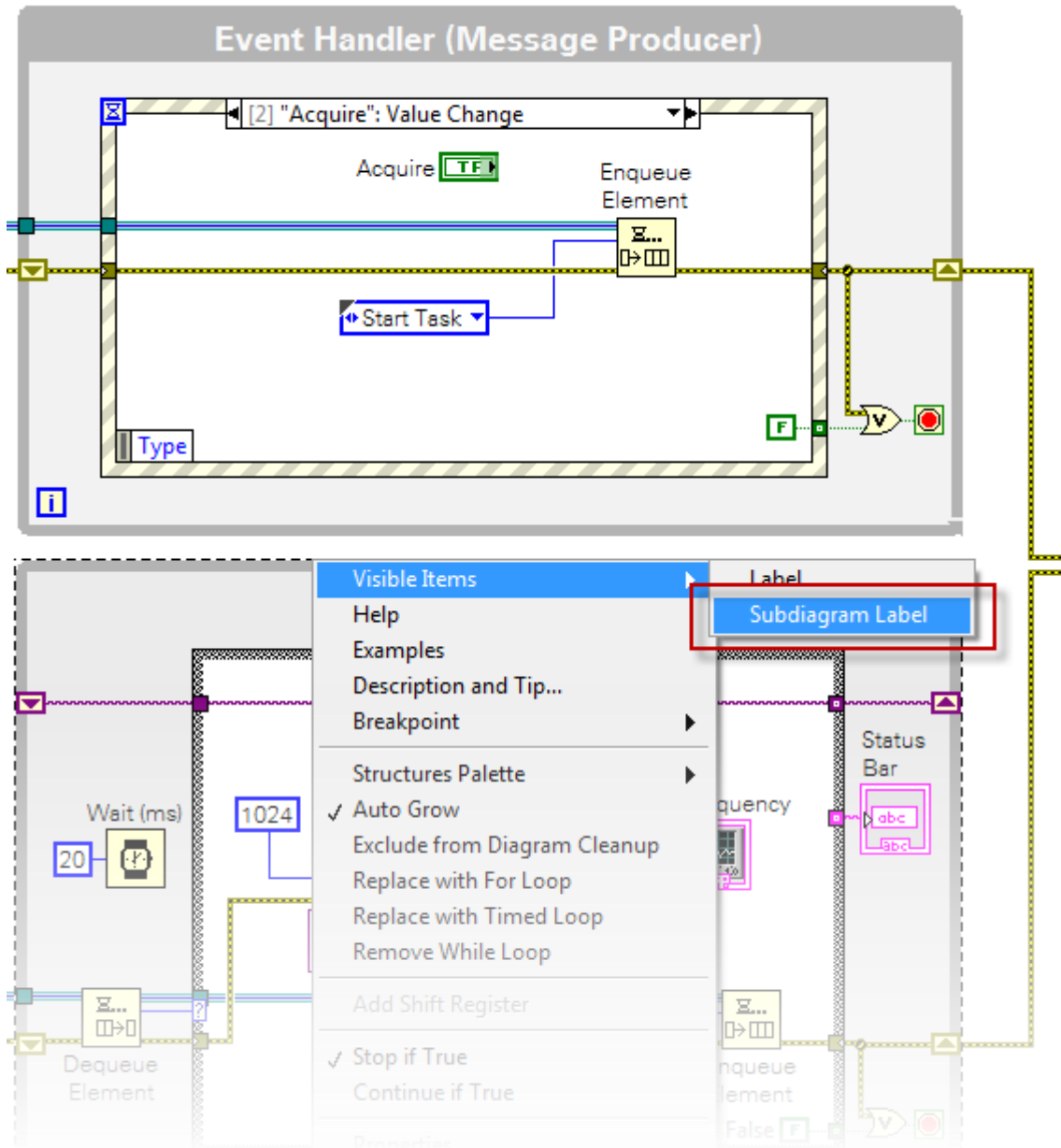
Many of the improvements streamline repetitive tasks or save multiple-clicks

CONCEPTS COVERED

- Subdiagram labels
- Right-click on multiple items menu options
- String editor dialog
- Conditional loop tunnel

Step 1: Subdiagram Labels

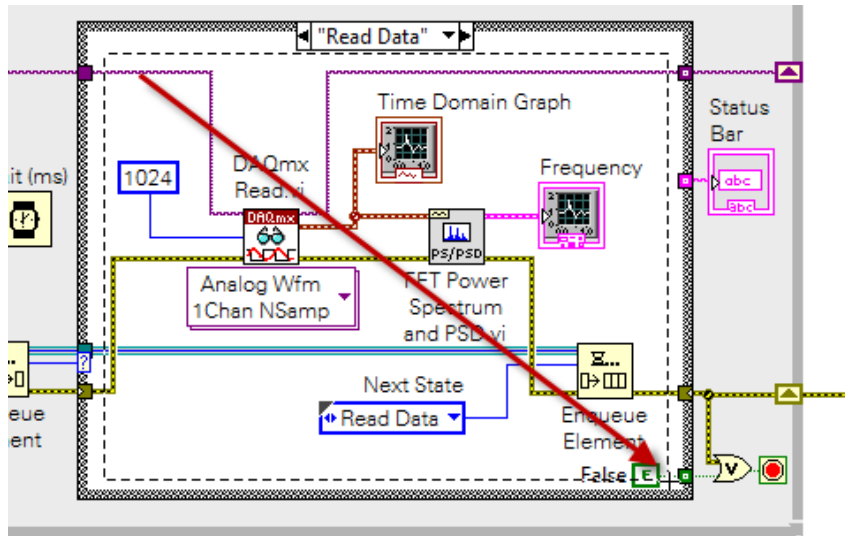
1. Expand the 'Productivity Enhancements' virtual folder
2. Open 'Instantaneous Measurement.vi'
3. Switch to the block diagram (CTRL + E)
4. Right click on the top while loop and select 'Visible Items > Subdiagram Label.' Repeat this process for the bottom loop. These Subdiagram labels are designed to move and resize with the loop, making it easier to keep diagrams clean and well-organized. (Note: these have already been colored to match the border. New structures will use the default color)



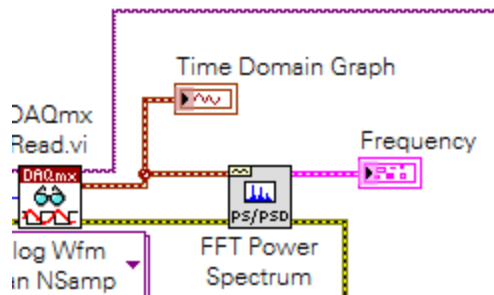
5. Expand the while loops to demonstrate that the Subdiagram labels automatically resize. (Undo size changes when finished)

Step 2: Right-click Menu Options

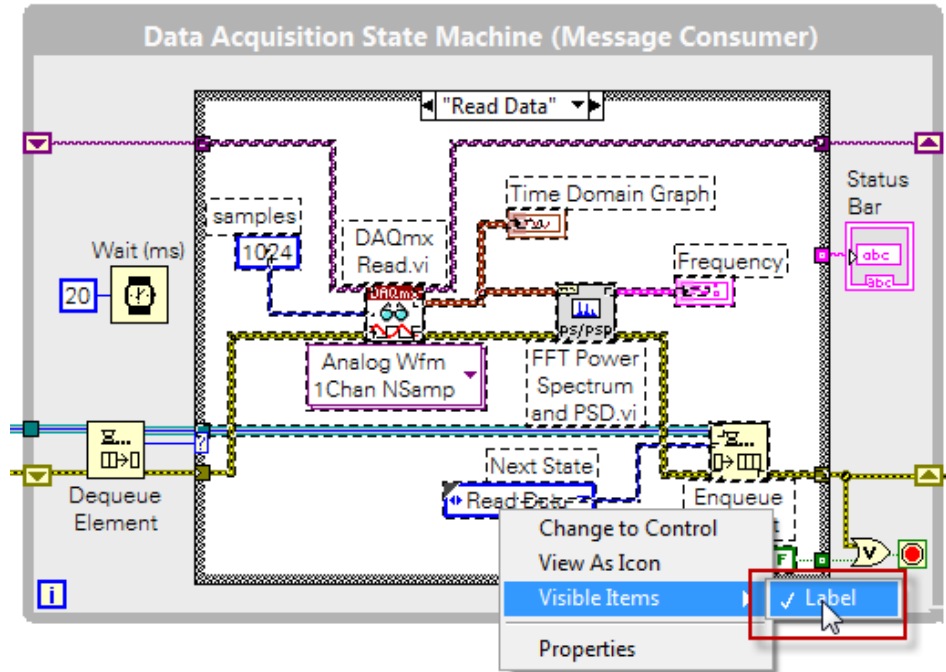
1. Make sure that the 'Read Data' case is showing in the bottom loop. Note that all of the terminals are currently being displayed as icons which takes up a lot of room on the block-diagram. To change these from icons to smaller terminals, we would normally have to perform this operation on each individual item.
2. Click and drag the mouse to select the contents of this case structure, as shown below



3. Right-click on the Frequency Graph Icon and de-select 'View as Icon.' Both the 'Time Domain Graph' and 'Frequency' icon should now display as terminals, as shown below.



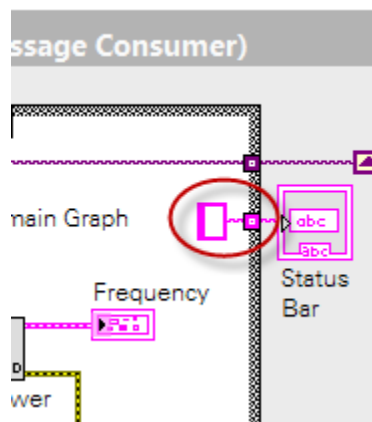
4. This is one of several operations that we can perform on multiple items simultaneously. As another example, we can right-click and toggle labels on and off. Ensure that the same items are still selected, right-click on the 'Next State' enumerated constant and de-select 'Visible Items > Labels,' as shown below.



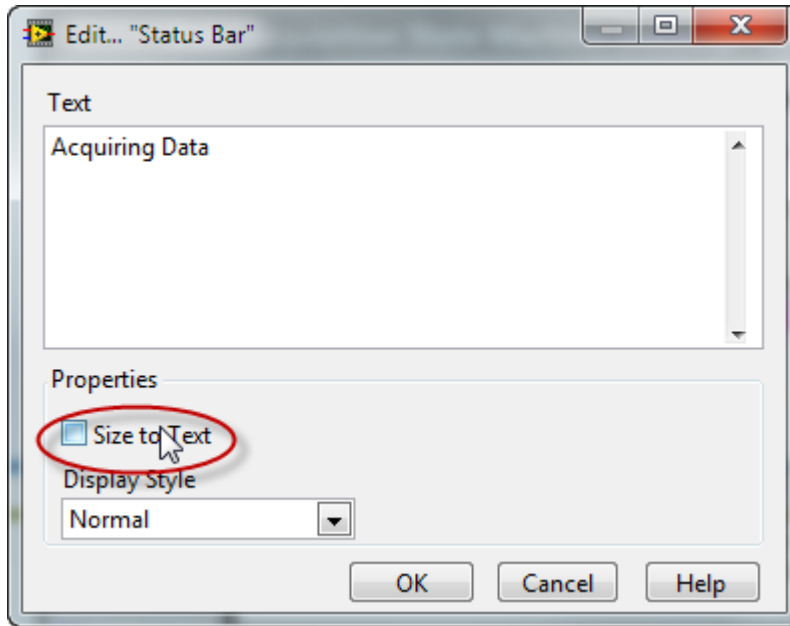
5. All of the items in this case should now be displayed without labels. This operation can be repeated to turn the labels back on.

Step 3: String Editing Dialog

1. Right-click on the string tunnel to the 'Status Bar' icon and select 'Create > Constant' to place a new string constant.



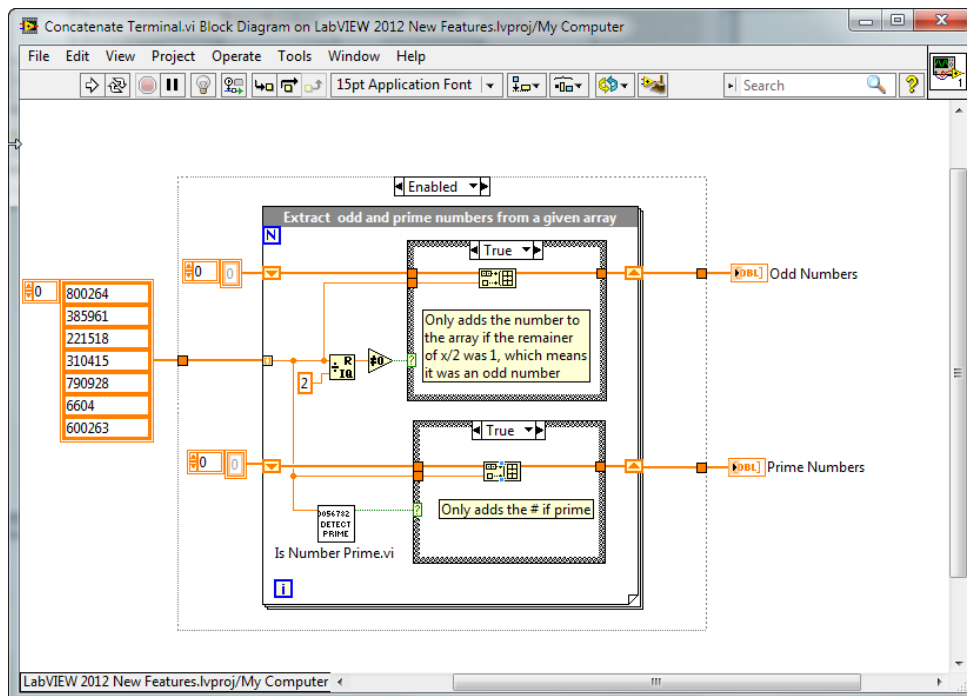
2. Right-click on the string constant and select 'Edit.' This will launch the new string editing dialog in LabVIEW 2012 .
3. When the 'Read Data' case is showing, we want it to display 'Acquiring Data,' so we'll type 'Acquiring Data' into the editor.
4. De-select 'Size to Text,' as shown below and click 'OK'



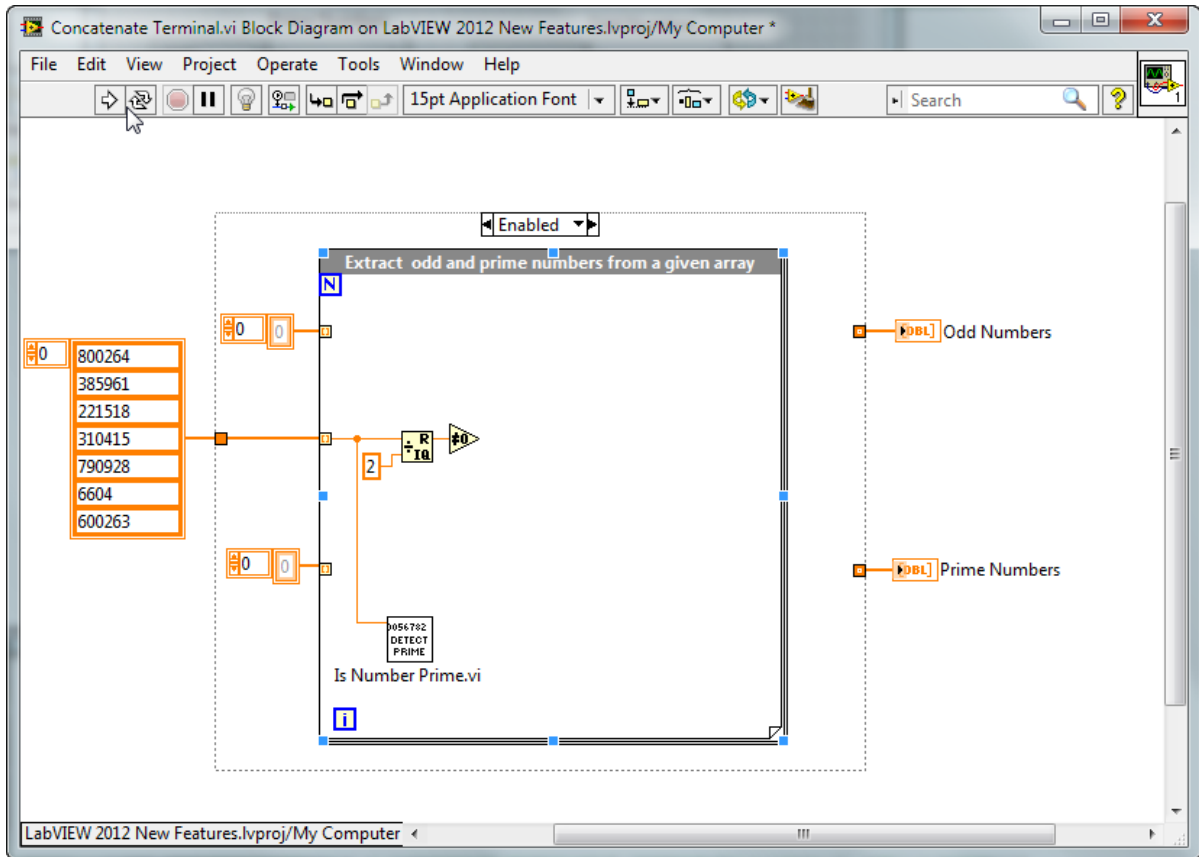
5. This text will now be stored in the constant, and can be edited without expanding the string constant on the block-diagram.
6. Switch to the front-panel and run the VI. Note that when we hit 'acquire' the VI's status indicator displays 'Acquiring Data.'

Step 4: Conditional Loop Tunnel

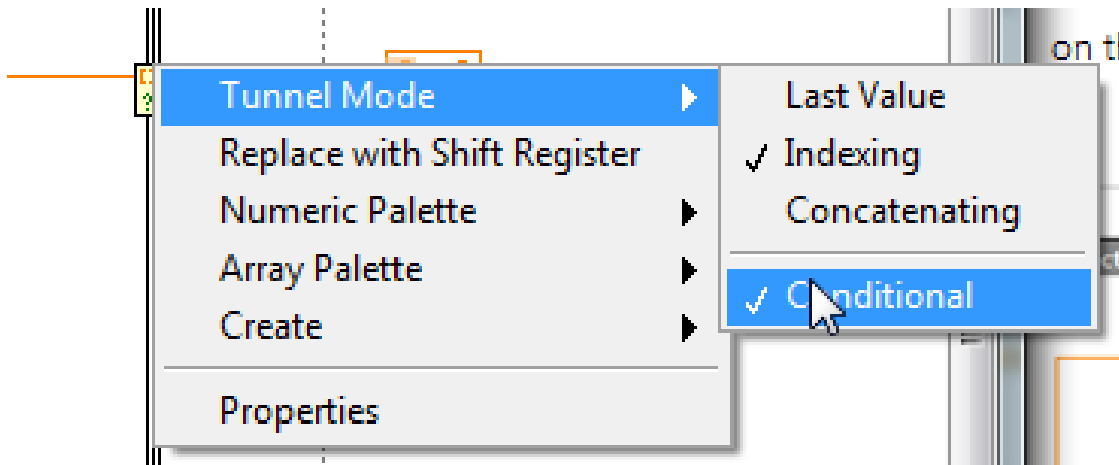
1. Return to the 'LabVIEW 2012 New Features' Project Explorer, expand the 'Productivity Enhancements from Idea Exchange' folder and open 'Concatenate Terminal.vi' and open the block diagram (CTRL + E).



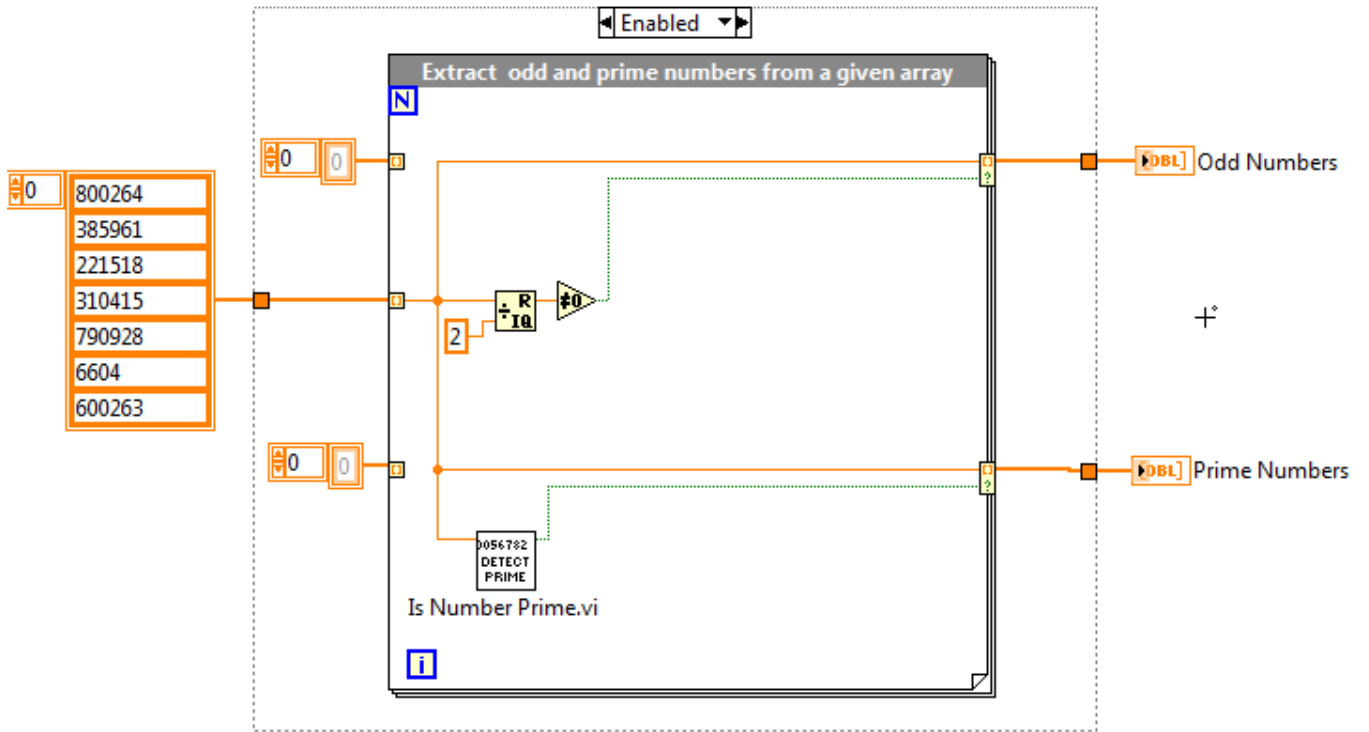
- This block diagram illustrates how to conditionally build array based upon a boolean criteria. This block-diagram is building two arrays: one for odd numbers, and the other for prime numbers, which previously required the use of the case structure and the build-array primitive (as shown).
- We can eliminate the case structure and simplify the code by using the conditional terminal. Remove both case structures by selecting them and pressing 'Delete.' Press CTRL + B to clean up the block diagram. The block diagram should now look like this:



- Wire the original array directly to the indicators, which will require an indexing tunnel. Right click on these tunnels and select 'Tunnel Mode > Conditional'



5. Wire the Boolean value of the 'Not Equal to 0' primitive and the output of 'Is Number Prime.vi' to the respective conditional tunnels. Now the code should look like this:
6. Run this VI – it is functionally equivalent to what we had earlier, but significantly simpler to code and understand.



3D STEREO VISION

GOAL

Use the new analysis capabilities within the Vision Development Module to determine object distance.

SCENARIO

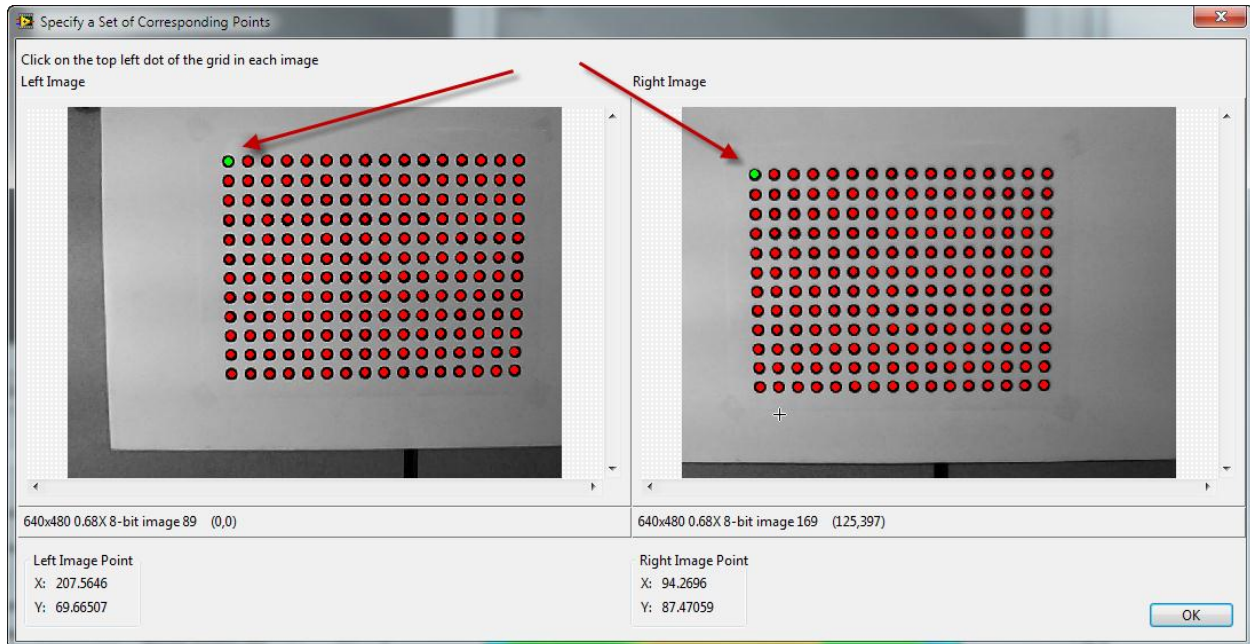
Images from two cameras can be used to create a depth-of-field map in LabVIEW thanks to new analysis and calibration capabilities. In this exercise, we will use images that were captured ahead of time.

CONCEPTS COVERED

- Calibration
- Depth Image
- Disparity Image

Step 1: Calibrate the Cameras

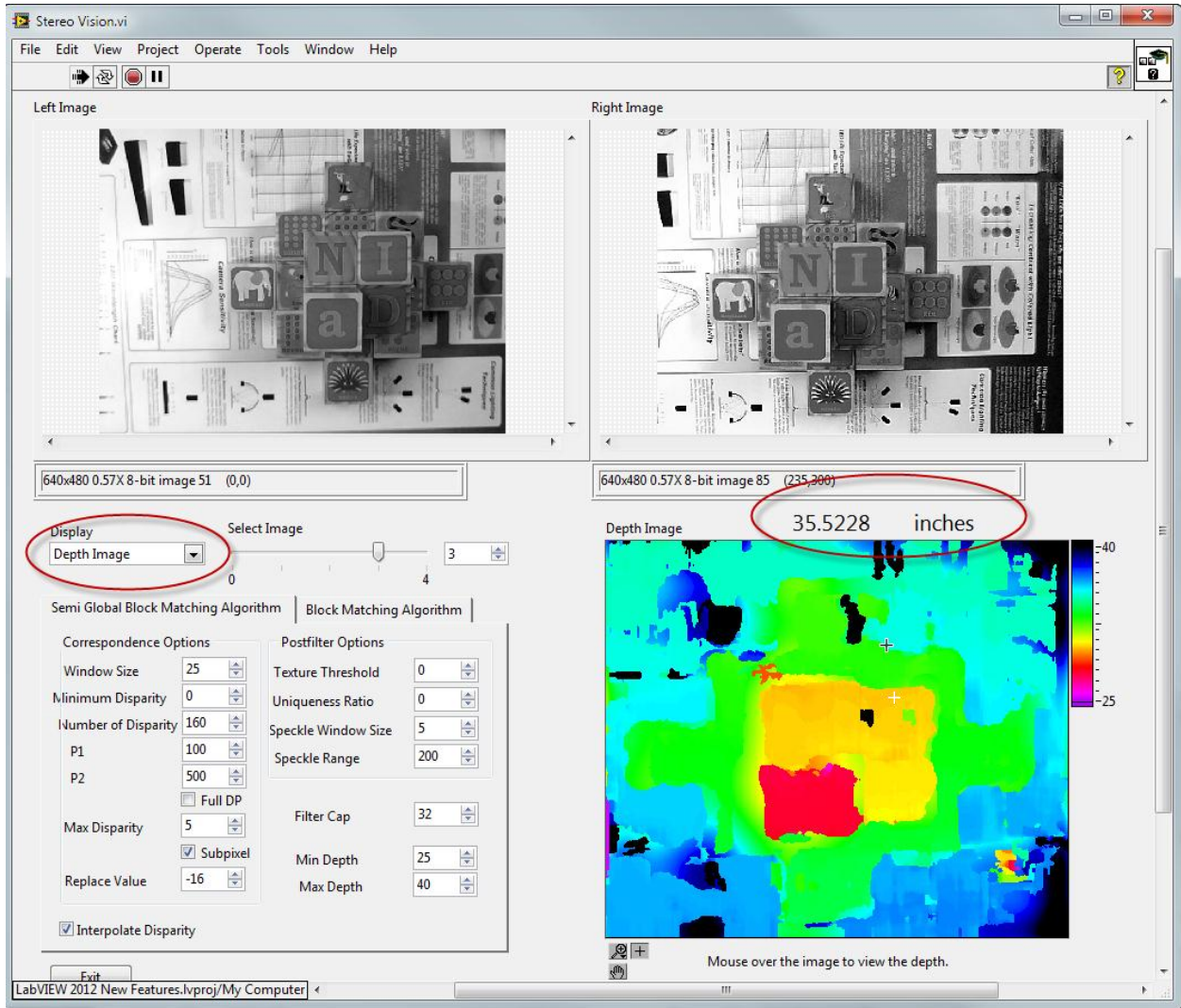
1. Expand the '3D Stereo Vision' virtual folder and launch 'Stereo Vision.vi'
2. Run the VI
3. Preloaded calibration images are analyzed to determine the depth of items being viewed.
4. When the two images are displayed, select the top left red dot on both to complete the calibration.



5. Click OK to save the calibration.

Step 2: Calculate Depth Map and Disparity Map

1. The Display drop down should default to 'Depth Image.'
2. The depth image should be displayed for the top two images in the lower right corner. Mouse over the image to determine the object's distance in that location.
3. Move the slider between 0 and 4 to display a different pair of images.
4. Change the drop-down to 'Disparity Image' to see the disparity map



5.

SPARSE MATRIX PERFORMANCE

GOAL

Improve the performance of complex analysis algorithms through the use of the LabVIEW Multicore Analysis and Sparse Matrix Toolkit.

SCENARIO

Compare the performance and memory use of the same operation when calculated using regular (dense) matrices versus sparse matrices.

CONCEPTS COVERED

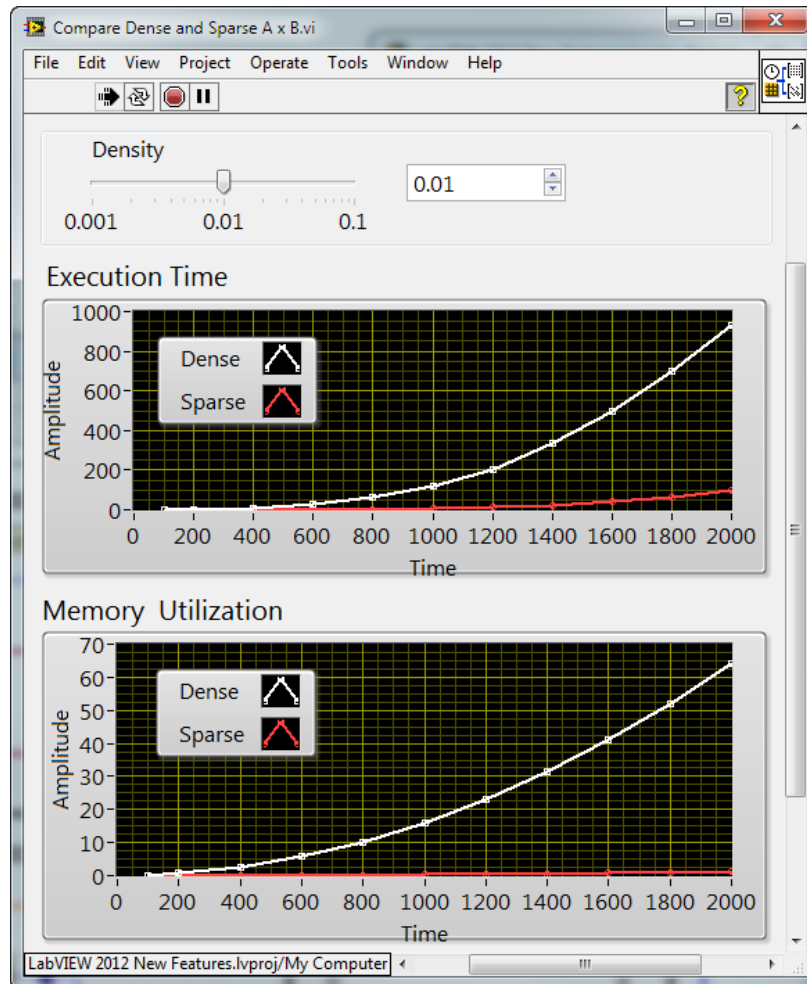
- New analysis palettes
- How to improve performance

SETUP

- Ensure the LabVIEW Multicore Analysis and Sparse Matrix Toolkit is installed

Step 1: Open and Run the Matrix Multiplication Example

1. Expand the 'Sparse Matrix Performance' virtual folder and launch the 'Compare Dense and Sparse A x B.vi'
2. Ensure the slider is set to .01 for the matrix density.
3. Switch to the block diagram and run the VI – keep in mind that this is a processor intensive operation that will take approximately 25 seconds on most machines.
4. While the VI is running, observe that we are compare the time required to execute two operations.
5. Press CTRL+H to activate contextual help and hover over the orange wire. The contextual help should confirm that this is a normal dense matrix data-type.
6. Hover over the red wire and observe that contextual help indicates that this is a sparse matrix data-type.
7. When the operation finishes, switch to the front panel and observe the comparison for performance and execution speed. The use of the sparse matrix operations improves performance for especially large datasets and keeps memory usage consistent.



MORE INFORMATION

ONLINE RESOURCES

- ni.com/labview/whatsnew