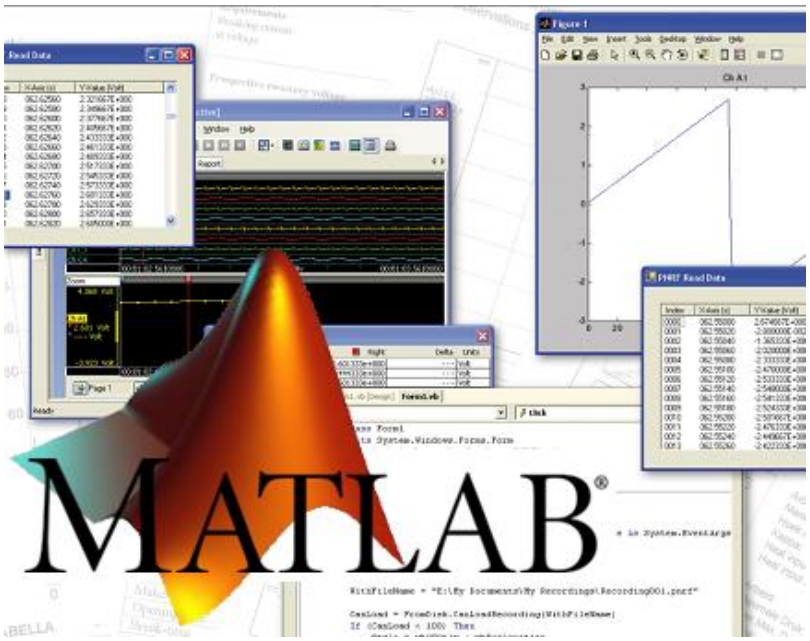
Genesis HIGH SPEED

User Manual

English



Perception Remote Control using MATLAB® Examples

A05412_01_E00_00

HBM

Document version 1.0 – June 2019

HBM GmbH
Im Tiefen See 45
64293 Darmstadt
Germany
Tel: +49 6151 80 30
Fax: +49 6151 8039100
Email: info@hbm.com
www.hbm.com/highspeed

LICENSE AGREEMENT AND WARRANTY

For more information about LICENSE AGREEMENT AND WARRANTY refer to:

www.hbm.com/terms

Table of Contents

1 Getting Started

Welcome to the Perception Remote Control using MATLAB®.examples document. This document contains a couple of examples showing how you can remotely control Perception from within MATLAB.
For more information on the RPC interface we refer to the help file **COMPerceptionInterfaces.chm** which can be downloaded from the HBM website.

1.1 Introduction

MATLAB® is a well-known language for technical computing. It integrates computation, visualization, and programming in an environment where problems and solutions are expressed in familiar mathematical notation.
MATLAB provides interfaces to clients or servers communicating via Component Object Model (COM). In this section we will describe how you can interface MATLAB to the Perception COM-RPC interface.

1.2 Intended audience

This documentation assumes you have sufficient knowledge of MATLAB®, this manual is NOT a tutorial on how to use MATLAB®.
This documentation also assumes you understand the HBM acquisition terminology. Understanding acquisition terminology is vital to understanding the Start, Stop, Pause recording commands and the manual Trigger command.

1.3 Requirements and installation

We assume you have installed MATLAB and the Perception software or the Perception COM-RPC in case you are running from a remote PC where Perception is not running.

2 Example 1 – Start Recording

In this example you will connect to Perception and start a recording. Before you can run the MATLAB code make sure that the Perception software program is running and connected to a GEN system or to one of the available simulators. To check if the Perception configuration is OK you only have to press the Start acquisition button and check if the recording starts. If this is the case your system is ready to be used by the MATLAB demo code. Do not forget to stop the recording in case you just started it for the above mentioned check.

```
% Create OLE Automation server to the Perception COM-RPC
Serv = actxserver('PerceptionCom.PerceptionCom');
% Set the server address to localhost, you can also use the IP address
Serv.SetServerAddress("localhost");
% Connect to Perception
Serv.ConnectToServer
% Start a recording
Serv.Start
```

3 Example 2 – Stop Recording

In this example you will connect to Perception and stop a running recording

```
% Create OLE Automation server to the Perception COM-RPC
Serv = actxserver('PerceptionCom.PerceptionCom');
% Set the server address to localhost, you can also use the IP address
Serv.SetServerAddress("localhost");
% Connect to Perception
Serv.ConnectToServer
% Stop a recording
Serv.Stop
```

4 Example 3 – Pause Recording

Shows how to pause a running recording

```
% Create OLE Automation server to the Perception COM-RPC
Serv = actxserver('PerceptionCom.PerceptionCom');
% Set the server address to localhost, you can also use the IP address
Serv.SetServerAddress("localhost");
% Connect to Perception
Serv.ConnectToServer
% Pause a recording
Serv.Pause
```

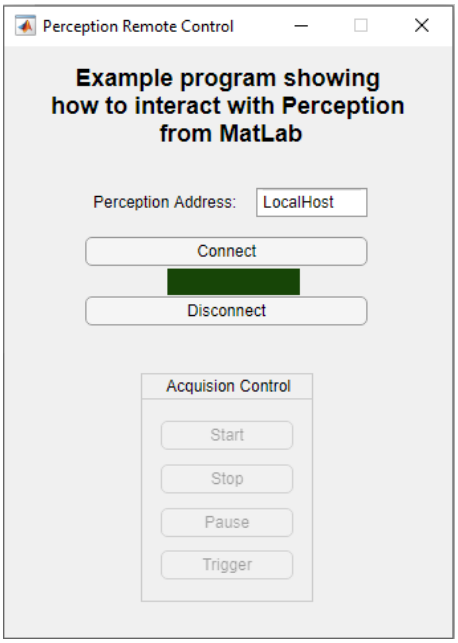
5 Example 4 – Manual Trigger

Shows how sent a manual trigger to Perception during a recording

```
% Create OLE Automation server to the Perception COM-RPC
Serv = actxserver('PerceptionCom.PerceptionCom');
% Set the server address to localhost, you can also use the IP address
Serv.SetServerAddress("localhost");
% Connect to Perception
Serv.ConnectToServer
% Sent a manual trigger to Perception
Serv.Trigger
```

6 Example 5 – GUI controlling acquisition state Perception

This demo is using the **MatLab App Designer** to create a small program which controls the Perception acquisition state.
The program GUI looks like:



The code behind this GUI looks like

```
classdef PerceptionMatLabDemo < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        PerceptionRemoteControlUIFigure  matlab.ui.Figure
        ConnectButton                matlab.ui.control.Button
        Title                        matlab.ui.control.Label
        PerceptionAddressEditFieldLabel  matlab.ui.control.Label
        PerceptionAddressEditField  matlab.ui.control.EditField
        AcquisionControlPanel        matlab.ui.container.Panel
        StartButton                  matlab.ui.control.Button
        StopButton                   matlab.ui.control.Button
        PauseButton                  matlab.ui.control.Button
        TriggerButton                matlab.ui.control.Button
        DisconnectButton             matlab.ui.control.Button
        ConnectionStatus             matlab.ui.control.Label
    end

    properties (Access = private)
        Serv = 0; % Perception Server object
        IsConnected = false;
    end

    methods (Access = private)

        function UpdateUIConnectionStatus(app)
            if (app.IsConnected)
               app.ConnectionStatus.BackgroundColor = [0.4667 0.9686 0.302];
               app.StartButton.Enable = true;
               app.StopButton.Enable = true;
               app.PauseButton.Enable = true;
               app.TriggerButton.Enable = true;
            else
               app.ConnectionStatus.BackgroundColor = [0.090196 0.270588 0.027451];
               app.StartButton.Enable = false;
```

```
            app.StopButton.Enable = false;
            app.PauseButton.Enable = false;
            app.TriggerButton.Enable = false;
        end
    end
end

% Callbacks that handle component events
methods (Access = private)

    % Code that executes after component creation
    function OnStartup(app)
        UpdateUIConnectionStatus(app);
    end

    % Button pushed function: ConnectButton
    function Connecting(app, event)
        % Create Automation server to the Perception RPC
        app.Serv = actxserver('PerceptionCom.PerceptionCom');
        % Set the server address to localhost, you can also use the IP address
        app.Serv.SetServerAddress(app.PerceptionAddressEditField.Value);
        % Connect to Perception
        app.Serv.ConnectToServer;
        app.IsConnected = true;
        UpdateUIConnectionStatus(app);
    end

    % Button pushed function: StartButton
    function StartRecording(app, event)
        app.Serv.Start;
    end

    % Button pushed function: StopButton
    function StopRecording(app, event)
         app.Serv.Stop;
    end

    % Button pushed function: PauseButton
    function PausePerception(app, event)
        app.Serv.Pause;
    end

    % Button pushed function: TriggerButton
    function TriggerPerception(app, event)
        app.Serv.Trigger;
    end

    % Button pushed function: DisconnectButton
    function Disconnect(app, event)
        app.Serv.DisconnectFromServer;
        app.IsConnected = false;
        UpdateUIConnectionStatus(app);
    end

    % Close request function: PerceptionRemoteControlUIFigure
    function OnClose(app, event)
        if (app.IsConnected)
            app.Serv.DisconnectFromServer;
        end
        delete(app);
```

```
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create PerceptionRemoteControlUIFigure and hide until all components are created
            app.PerceptionRemoteControlUIFigure = uifigure('Visible', 'off');
            app.PerceptionRemoteControlUIFigure.AutoResizeChildren = 'off';
            app.PerceptionRemoteControlUIFigure.Position = [400 400 338 447];
            app.PerceptionRemoteControlUIFigure.Name = 'Perception Remote Control';
            app.PerceptionRemoteControlUIFigure.Resize = 'off';
            app.PerceptionRemoteControlUIFigure.CloseRequestFcn = createCallbackFcn(app,
@OnClose, true);
            app.PerceptionRemoteControlUIFigure.Scrollable = 'on';

            % Create ConnectButton
            app.ConnectButton = uibutton(app.PerceptionRemoteControlUIFigure, 'push');
            app.ConnectButton.ButtonPushedFcn = createCallbackFcn(app, @Connecting, true);
            app.ConnectButton.Position = [62 282 213 22];
            app.ConnectButton.Text = 'Connect';

            % Create Title
            app.Title = uilabel(app.PerceptionRemoteControlUIFigure);
            app.Title.HorizontalAlignment = 'center';
            app.Title.FontSize = 18;
            app.Title.FontWeight = 'bold';
            app.Title.Position = [5 357 334 91];
            app.Title.Text = {'Example program showing '; 'how to interact with Perception ';
'from MatLab'};

            % Create PerceptionAddressEditFieldLabel
            app.PerceptionAddressEditFieldLabel =
uilabel(app.PerceptionRemoteControlUIFigure);
            app.PerceptionAddressEditFieldLabel.HorizontalAlignment = 'right';
            app.PerceptionAddressEditFieldLabel.Position = [62 319 114 22];
            app.PerceptionAddressEditFieldLabel.Text = 'Perception Address:';

            % Create PerceptionAddressEditField
            app.PerceptionAddressEditField = uieditfield(app.PerceptionRemoteControlUIFigure,
'text');
            app.PerceptionAddressEditField.Position = [191 319 84 22];
            app.PerceptionAddressEditField.Value = 'LocalHost';

            % Create AcquisionControlPanel
            app.AcquisionControlPanel = uipanel(app.PerceptionRemoteControlUIFigure);
            app.AcquisionControlPanel.AutoResizeChildren = 'off';
            app.AcquisionControlPanel.TitlePosition = 'centertop';
            app.AcquisionControlPanel.Title = 'Acquision Control';
            app.AcquisionControlPanel.Position = [104 28 130 173];

            % Create StartButton
            app.StartButton = uibutton(app.AcquisionControlPanel, 'push');
            app.StartButton.ButtonPushedFcn = createCallbackFcn(app, @StartRecording, true);
            app.StartButton.Enable = 'off';
            app.StartButton.Position = [15 115 100 22];
            app.StartButton.Text = 'Start';
```

```
            % Create StopButton
            app.StopButton = uibutton(app.AcquisionControlPanel, 'push');
            app.StopButton.ButtonPushedFcn = createCallbackFcn(app, @StopRecording, true);
            app.StopButton.Enable = 'off';
            app.StopButton.Position = [15 82 100 22];
            app.StopButton.Text = 'Stop';

            % Create PauseButton
            app.PauseButton = uibutton(app.AcquisionControlPanel, 'push');
            app.PauseButton.ButtonPushedFcn = createCallbackFcn(app, @PausePerception, true);
            app.PauseButton.Enable = 'off';
            app.PauseButton.Position = [15 49 100 22];
            app.PauseButton.Text = 'Pause';

            % Create TriggerButton
            app.TriggerButton = uibutton(app.AcquisionControlPanel, 'push');
            app.TriggerButton.ButtonPushedFcn = createCallbackFcn(app, @TriggerPerception,
true);
            app.TriggerButton.Enable = 'off';
            app.TriggerButton.Position = [15 17 100 22];
            app.TriggerButton.Text = 'Trigger';

            % Create DisconnectButton
            app.DisconnectButton = uibutton(app.PerceptionRemoteControlUIFigure, 'push');
            app.DisconnectButton.ButtonPushedFcn = createCallbackFcn(app, @Disconnect, true);
            app.DisconnectButton.Position = [62 237 213 22];
            app.DisconnectButton.Text = 'Disconnect';

            % Create ConnectionStatus
            app.ConnectionStatus = uilabel(app.PerceptionRemoteControlUIFigure);
            app.ConnectionStatus.BackgroundColor = [0.0902 0.2706 0.0275];
            app.ConnectionStatus.Position = [124 260 100 20];
            app.ConnectionStatus.Text = '';

            % Show the figure after all components are created
            app.PerceptionRemoteControlUIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = PerceptionMatLabDemo

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.PerceptionRemoteControlUIFigure)

            % Execute the startup function
            runStartupFcn(app, @OnStartup)

            if nargout == 0
                clear app
            end
        end
```

```
        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.PerceptionRemoteControlUIFigure)
        end
    end
end
```

Head Office
HBM
Im Tiefen See 45
64293 Darmstadt
Germany
Tel: +49 6151 8030
Email: info@hbm.com

France
HBM France SAS
46 rue du Champoreux
BP76
91542 Mennecy Cedex
Tél:+33 (0)1 69 90 63 70
Fax: +33 (0) 1 69 90 63 80
Email: info@fr.hbm.com

Germany
HBM Sales Office
Carl-Zeiss-Ring 11-13
85737 Ismaning
Tel: +49 89 92 33 33 0
Email: info@hbm.com

UK
HBM United Kingdom
1 Churchill Court, 58 Station Road
North Harrow, Middlesex, HA2 7SA
Tel: +44 (0) 208 515 6100
Email: info@uk.hbm.com

USA
HBM, Inc.
19 Bartlett Street
Marlborough, MA 01752, USA
Tel : +1 (800) 578-4260
Email: info@usa.hbm.com

PR China
HBM Sales Office
Room 2912, Jing Guang Centre
Beijing, China 100020
Tel: +86 10 6597 4006
Email: hbmchina@hbm.com.cn

measure and predict with confidence

A05412_01_E00_00