Version

2.1

WATSON

Head Tracking and Gesture Recognition Library

User Guide

WATSON

User Guide

Louis-Philippe Morency
lmorency@csail.mit.edu

Version 2.1a

August 17th 2006

Table of Contents

Copyright

Chapter

1

Introduction

Watson is a keyframe-based tracking library that uses stereo or monocular images to track the position and orientation of a rigid object.

This guide presents the functionalities and characteristics of Watson tracking library[1]. Watson has been originally created to estimate the position and orientation of the head using a stereo camera but the current version also work with monocular cameras. The current version can track for a long period of time with bounded drift the 6 degrees-of-freedom of the head. Also, the tracker can be reconfigured to estimate the pose of any rigid object and to estimate ego-motion when the background is static.

To get good precision and reduce the possible drift, Watson implements Adaptive View-based Appearance Models technique described in [2] which acquires keyframes of the object online during the tracking. These keyframes represent the object in different pose. When the trajectory of the object crosses one of the recorded keyframe, the pose estimation algorithm will take in account the pose of the keyframe. The pair-wise pose estimation is done using a hybrid technique [3] that combines Iterative Closes Point and Normal Flow Constraint. The complete system can track object for a long period of time with bounded drift.

The following chapter explains the installation procedure for the tracking library. Chapter 3 explains the different parameters of the software. Chapter 4 presents the network protocol used to communicate with Watson via TCP/IP sockets.

Chapter 2

Installation

Watson can be easily installed on a Microsoft Windows system using the IntallShield installation package. After running the setup, you will need to calibrate you camera if you want to use the tracking system in real-time.

Setup Programs

Main setup

The core installation file watson-x.xx.exe will copy on your machine the following components:

- Watson\bin: Watson demo program (Watson.exe) and DLLs necessary to run the application;
- Watson\Classifier: Features for the frontal and side-view face detectors as well as the eye detectors;
- Watson\HMMs: Learned Hidden Markov Models (HMMs) for head nods and head shakes detection;
- Watson\SVMs: Learned Support Vector Machines (SVMs) for head nods and head shakes detection;
- Watson\EigenSpaces: Learned eigen spaces for eye gaze estimation;
- Watson\include: Include files for the C++ interface;
- Watson\lib: Libraries for the C++ interface;
- Watson\Samples: Samples programs for Watson C++ interface;

- Watson\Sequences\SRI: Configuration files for running the tracker online with a Videre Design stereo camera.

- Watson\Sequences\USB: Configuration files for running the tracker online with a USB monocular camera.

- \Watson\Sequences\ExempleStereo: Pre-recorded stereo video sequence. This sequence can be used to test if the installation of the demo program has been done correctly or to run the tracker with different settings.

- \Watson\Sequences\ExempleAVI: Pre-recorded monocular video sequence. This directory shows how to use Watson to track the head position and orientation from a AVI movie file

Before to be able to run Watson online (directly from a stereo camera), you will need to setup your stereo camera and calibrate it. The following section explains how to do it when you are using a Videre Design stereo camera.

Libraries installed

Watson has been coded to take takes advantage of the MMX and SSE capacity of the Pentium 3 and Pentium 4. When you install the demo program, different DLLs are copied in the \Watson\bin directory

- Intel Integrated Performance Primitives 5.1
- Intel Math Kernel Library 8.1.1
- Intel Open Source Computer Vision Library 1.0
- Small Vision System 4.3a
- GLUT 3.7

If you already installed one of those libraries on your computer and have problem to run Watson, you should check your PATH variable to be sure that there is no conflict between different versions.

We use Qt as our GUI interface because of its speed, simplicity and compatibility with Linux (and now Macintosh too!). We use the version 3.2.3 of Qt. For 3D display we use OpenGL and its extension, GLUT 3.7.

Software updates

When you are updating Watson, most files will be replaced with the newest version. For the sub-directories of \Watson\Sequences, only the files ParamWatson.cfg will be updated. For this reason, you should put all your personalized parameters in ParamWatsonUser.cfg. This

way you will be able to use the latest default parameters from ParamWatson.cfg but keep your personalized parameters.

Monocular Camera Calibration

USB camera

Starting with version 2.0, Watson can now track the head pose using a normal USB webcam. To optimize the tracking, some parameters most be set in the control panel of your USB camera. The most important setting to change is the automatic brightness and gain parameters. This parameter should be turned off so that the brightness doesn't change during the recording. This will improve the performance of the optical flow estimation during tracking.

Stereo Camera Calibration

Videre Design

The tracking system has been extensively tested with the Mega-D stereo camera from Videre Design. Recently, the system has been modified to handle the new DCS model from the same company. The following paragraphs will give you some guideline on how to setup the cameras, for more information, please refer to the user guide installed with the Small Vision System.

The first step is to install the Small Vision System (SVS) using the setup file svs42d.exe.. To be able to install the library, you will need a valid license number (please contact Videre Design if you don't have it). It is preferable to install the library before you plug the stereo camera for the first time since the driver of the camera is installed with the SVS library. When you plug your camera in your firewire card the driver setup should start automatically.

When SVS is installed, you need to specify which type of camera you have. If you have a Mega-D, you should run the batch files \svs42\bin\setup_megad.bat and Start->Programs->Watson->Setup cameras->MegaD. If you have a DCS, you should run the batch files \svs\bin\setup_dcs.bat and Start->Programs->Watson->Setup cameras->DCS. Now you are ready to calibrate your camera.

To start the calibration, run the program \svs\bin\smallvcal.exe. If your installation worked correctly, you should be able to start grabbing images by setting the Input to Video and pressing the button Continuous. You should see the left and right image displayed. Now press the menu button Calibrate… to start the calibration. The software uses 10 images of a check board to estimate the intrinsic and extrinsic parameters of the camera. Please read the SVS manual for details on the calibration procedure. When the calibration is done, save your calibration file in the directory \Watson\Sequences\SRI\. A good name for the calibration file is calib-xxxxx.ini where xxxxx represents the serial number of your camera.

The last step before to be able to grab directly from the stereo camera using Watson demo program, is to modify the parameters files of Watson so it use your new calibration file. To do so, open the file \Watson\Sequences\SRI\ParamWatsonUser.cfg, search the field CONFIG_FILENAME: and modify its value to be the name of your calibration file (calib-xxxxx.ini). Now you are ready run Watson directly from your stereo camera!

Other stereo cameras

Watson tracking system has been also tested on Digiclops stereo cameras but unfortunately the demo program (Watson.exe) can only run with Videre Design stereo cameras. To use Watson tracking library with other stereo camera, please look at the C++ interface described in Chapter 6.

Chapter

3

Software Interface

The demo program gives you a visual interface to test different settings of Watson library. It gives to the user the flexibility to change most tracking parameters. The tracker can be run online, using images directly from the camera, or offline, using images from a prerecorded sequence stored on disk.

Main software functionalities

Watson tracking system can be run online or offline. When the program starts, Watson automatically looks in the current directory for two parameter files: ParamWatsonUser.cfg and ParamWatson.cfg. These files, as described in the following chapter, contain all the default and user-defined parameters necessary for running the tracker. Watson should be started in a directory where ParamWatson.cfg (ParamWatsonUser.cfg is optional) is present.

Grabbing and Tracking

Watson automatically switch between grabbing and tracking when the AutoInit (CTRL+A) option is activated. To start continuous grabbing press F2 and to stop it press F4. With AutoInit activated, as soon as a face is detected in the image, the tracker will start. At each time step a frame is grabbed, segmented and finally the pose of the object is estimated.

Load Sequence

To load a prerecorded sequence from the demo program, you can select Load Sequence in the Files menu and click on the "ParamSeq.cfg" representing the sequence you want to load (for example \sequences\ExempleStereo\ParamSeq.cfg). This file contains all the calibration information for the sequence. As explained in the next chapter, this process can be automated by modifying the ParamWatsonUser.cfg to point on a specific ParamSeq.cfg.

Output console

The output console gives you some information about the pose estimate (rotation and translation) of the object as well as the results from the head nods detector. The top frame

shows the absolute pose of the object. The translation is displayed in millimeters and the rotation is displayed in degrees. The variance gives an idea of the accuracy of the pose. The middle frame represents the displacement between the previous frame and the current frame (velocity of the object). The third frame shows the approximate center of the object (also in millimeters). The last frame shows the results from the head nods and headshakes detectors. The numbers below each button represent the confidence of each detector.

Position, Orientation and Coordinate system

The referential coordinate system is set on the left camera for stereo cameras. It is a right-handed coordinate system where the Z axis point behind the camera, the Y axis point below the camera and the X axis point on the left side (when looking at the camera).

The position returned by the tracker represents the distance between the center of the object and the center of the left camera. The orientation returned by the tracker represents the rotation between the first tracked frame and the current frame. When using the Auto-initialization option, the tracker will start only if it finds a frontal face. Since the first tracked frame is a frontal view, each following frame will be relative to the frontal view.

To compute the absolute orientation of the object, you must apply the rotation [rx, ry, rz] to the initial orientation (frontal view: [0,0,-1]). The rotation notation used by Watson is based on a rotation axis and a rotation around this axis. The norm of the vector a=[rx, ry, rz] represents the amount of rotation in radian. The normalized vector represents the axis of rotation. You can change the notation to a rotation matrix by applying this equation $[R] = [I] + \sin(angle)[\sim axis] + (1-\cos(angle))[\sim axis]^2$ (see [4] for more details). Finally, the absolute orientation can be computed by applying the rotation matrix to the frontal view: orientation = [R]*[0,0,-1].

Parameter console

The parameter console gives a visual interface for most of the parameters of the tracker. You can find a description of those parameters in the following chapter. Also, the complete list of parameters can be found in the file `ParamWatson.cfg`.

Main shortcut keys

F2 - Start continuous grabbing/tracking (also on the toolbar);

F3 – Start continuous grabbing/tracking and record images on disk(also on the toolbar);

F4 - Stop grabbing/tracking (also on the toolbar);

F5 - Show current intensity image;

F6 - Show current depth image;

F7 – Show keyframe intensity image;

F8 – Show keyframe depth image;

CTRL+ 1 - Switch to No Display mode (no OpenGL display);

CTRL+ 2 - Switch to 2D mode;

CTRL+ 3 - Switch to 3D mode;

CTRL+ 0 - Switch between 3D modes: Frontal view or Top View;

CTRL+ A – Activate/deactivate the autoinitialization;

CTRL+P – Show/hide the Parameter console;

CTRL+O – Show/hide the Output console;

CTRL+L – Load a new sequence;

CTRL+R – Reload the current sequence;

Chapter 4

Parameter Files

The demo program gives you a visual interface to test different settings of Watson library. It gives to the user the flexibility to change most tracking parameters.

Files description

The tracking parameters are kept in 3 different files: ParamWatsonUser.cfg, ParamWatson.cfg and ParamSeq.cfg (or ParamSeqDirect.cfg). ParamWatson.cfg contains the default parameters for the tracker as well as some parameters for the display. You should not modify this file directly since your changes will lost next time you update Watson. Instead, you should enter the parameters you want to modify inside ParamWatsonUser.cfg since this file is never updated by the Installer. ParamSeq.cfg contains all the parameters relative to the grabbing. The parameter files are separated by sections:

- [SECTION_WATSON]: This is the main section of the parameter files. It sets some high-level parameters and specifies the path of other parameter files like ParamSeqDirect.cfg.
- [SECTION_NETWORK]: Set the networking options (client and server) of the demo program.
- [SECTION_HEAD_NODS]: This section sets parameters related to the HMMs (Hidden Markov Model) and SVMs (Support Vector Machines) trained for head nods and head shakes detection.
- [SECTION_MAP_BUILDER]: This section sets the parameters for the keyframes acquisition process. You can set how those keyframes will be acquired (tessellation or clustering) and the gap between each acquired keyframe.
- [SECTION_TRACKER_DIRECTOR]: This section specifies which tracker is activated, sets some main tracking parameters (MATCH_FUNCTION: and UPDATE_POSE:) and let you print some debug information like poses, velocity and center of mass.

- [SECTION_TRACKER_ICP]: Detailed parameters for the default tracker (ICP). Those parameters should be only changed by "advanced" users.
- [SECTION_INIT_TRACKER]: Set some parameters for the tracking initialization and reinitialization.
- [SECTION_SIMPLE_TRACKER]: This section sets parameters for the image segmentation. The setting of those parameters will influence the tracking initialization since only segmented pixels will be used for initialization.
- [SECTION_3D_MODEL]: This section sets parameters for the ellipse matching algorithm used during monocular tracking.
- [SECTION_RECORDER]: Set the default values for the recording option.
- [SECTION_OPEN_GL]: Set the display options of the demo program.
- [SECTION_SEQUENCE]: This section, found usually in ParamSeq.cfg or ParamSeqDirect.cfg, specifies the parameters related to the grabbing/stereo process. Some parameters like SIZE_ROI are used for tracking/segmentation purpose.
- [SECTION_FILES_GRABBER]: This section is used for pre-recorded sequences. It gives all the details about the file format and the camera used to record that sequence. You will usually find this file in ParamSeq.cfg.

Network parameters

The main way to communicate with Watson is via network. All the parameters related to networking are usually set in the section [SECTION_ NETWORK] of ParamWatsonUser.cfg. Watson supports 2 mode of communication: UDP (datagram) or TCP (socket). Also, Watson can be used as a client, a server or both. In the client mode, Watson can send information about the tracking results as well as the grabbed images. In the server mode, Watson receives the images from the network instead of grabbing them from camera or files.

Client mode

When setting up Watson in the client mode, you have to specify three kind of information:

- Which **information** do you want to be sent via network?
- In which **format** do you want the information?
- Which **host** will receive the information?

Currently, Watson can open up to 2 connections. This feature makes it possible to send the results of the head nod detector to one computer while sending the results of the head pose tracker to another computer. The parameter CONNECT_SOCKET: activate/deactivate the connection number 1 and the parameter CONNECT_SOCKET2: activate/deactivate the connection number 2.

Information parameters

The parameter TYPE_INFO_SENT: (or TYPE_INFO_SENT2:) specify which type of information will be sent to the connected computer. After the TYPE_INFO_SENT: tag, you should enumerate all the information tag you want. Each tag must be separated by a space and the line must end by the tag END. Here is a list of information tags available:

- INFO_LINKS: The details of each transformation computed during the tracking will be sent (see Section 5 for more details about the format).
- INFO_PREVIOUS_LINKS_ONLY (can't be used with INFO_LINKS): Equivalent to the velocity. This tag will send the transformation between each consecutive frame (see Section 5 for more details about the format).
- INFO_POSES: Send the absolute pose of the head for each frame.
- INFO_SCREEN_COORDS: Send the estimated projection of the "nose" on the screen. This option can be useful for moving the mouse cursor with your head. The screen is supposed to be parallel to the camera. The parameter SCREEN_POSITION: should be set adequately.
- INFO_CENTERS: The estimated center of mass of the object (in millimeters).
- INFO_HEAD_NODS: Send the results form the head nods detector.
- INFO_FRAME: Send Intensity image, depth image and frame info.
- INFO_INTENSITY: Send Intensity image only.
- INFO_DEPTH: Send Depth image only.

Also, the parameter `SEND_MESSAGE_DURING_TRACKING_ONLY:` can be set to `TRUE` or `FALSE` depending if you want to always receive network message (FALSE) or receive network messages only when the tracking is working (TRUE).

Format parameters

Each message sent via network is in ASCII format (at the exception of the images). A header is sent before each message to specify which information will follow. To define the format of those headers, you can use of those parameters:

- MESSAGE_PREFIX: Prefix used by every message (including images). This can be used to identify the information coming from Watson.
- MESSAGE_LINKS_SUFFIX: This parameter specifies which text should follow the MESSAGE_PREFIX: when a transformation is sent via network.

- MESSAGE_POSES_SUFFIX: This parameter specifies which text should follow the MESSAGE_PREFIX: when a pose is sent.
- MESSAGE_NODS_SUFFIX: This parameter specifies which text should follow the MESSAGE_PREFIX: when a head nods and head shakes detection results are sent via network.

Host parameters

Three parameters should be set to specify the address of your host and the type of connection:

- SOCKET_TYPE: (or SOCKET_TYPE2:) Can be TCP (for socket connection) or UDP (for datagram or connection-less).
- PORT_CONNECTION: (or PORT_CONNECTION2:) This specify the port for connection. Should be the same as your server (listener).
- NAME_HOST: (or NAME_HOST2:) This specify the name of your host. The name can be an IP address (xxx.xxx.xxx.xxx) or a machine name (registered on the DNS server).

Server mode

Watson can receive stereo images from a TCP/IP connection. To activate this option, you must set the parameter CONNECT_SERVER: TRUE. The parameters PORT_SERVER: and NAME_SERVER: set the name of the client that will connect to Watson. Please refer to section 5 for more details on the image format.

Remote commands

Starting with version 1.4, you can now remotely start and stop Watson. To do this, you must connect to Watson (Client or Server mode) and send one of the following commands:

- REINIT: Used to start or restart the tracker. This command is equivalent to the keyboard shortcut F2.
- RECORD: Used to start or restart the tracker and record images on the hard disk. This command is equivalent to the keyboard shortcut F3.
- STOP: Used to stop the tracker. This command is equivalent to the keyboard shortcut F4.

Each command name can be personalized using a prefix common to every commands and a suffix specific to each command. The parameter COMMAND_PREFIX: sets the common prefix to every command. By default, this parameter is an empty string. The parameters COMMAND_REINIT_SUFFIX, COMMAND_RECORD_SUFFIX and COMMAND_STOP_SUFFIX specify the suffix string for each command.

Chapter 5

Network Interface

Watson demo program gives you a bi-directional network interface to send images, change tracking parameters or gather tracking results.

Server vs. Client

Watson demo program can receive and send information at the same time. Usually, the information received would be tracking parameters, action commands like "Start Tracker" or stereo images grabbed by another program. The information sent by Watson will usually be tracking results like the head position and orientation, its velocity or the head nods and shakes detection results. The current supported formats for network communication are UDP (datagram) or TCP/IP sockets.

Client Protocol

When Watson acting as a client, the demo program will connect to a TCP/IP server (or connectionless, UDP) and start sending information via the socket. If the connection to the server is not initiated at the beginning, it will try to reconnect everytime a image is grabbed. The name of the server, the type of the connection (UDP or TCP/IP) and the type of information sent are all set in ParamWatson.cfg (please refer to chapter 4 for more details). Two type of information can be sent: tracking results and stereo images.

Stereo images transfer

It is possible to use Watson to grab stereo images and send images to a remote system (which could be another instance of Watson) via network. Please refer to the section on Server Protocol for more details about the stereo images format.

Tracking results transfer

After processing each new frame, Watson can optionally send the tracking results via a TCP/IP socket (or UDP datagram). As described in Chapter 4, Watson can send 3 types of

tracking results: poses, links, and head nods detection results. All the information sent on the socket will be in ASCII format.

Links format

A link represents the relative pose between two frames. During tracking, Watson computes two kinds of links: link between 2 consecutive frames and link between the current frame and a keyframe. As described in chapter 4, Watson can send all the links or only the consecutive links (also called previous link). Each link is sent using the following format:

```
[LinkTag] [I1] [I2] [var] [tx] [ty] [tz] [rx] [ry] [rz]
```

where

- `[LinkTag]`: Tag sent at the beginning of each link message. This tag can be customized in the parameter file (see chapter 4).
- `[I1]` : Index of the previous frame
- `[I2]` : Index of the current frame
- `[var]`: Variance of the link
- `[tx],[ty],[tz]`: Translation between the previous frame and the current frame (in mm)
- `[rx],[ry],[rz]`: Rotation between the previous frame and the current frame (in rad)

See Chapter 3 for more details on the position and orientation format.

Poses format

The pose represents the position and orientation of the object in a given frame. The pose information sent via network has the following format:

```
[PoseTag] [index] [variance] [tx] [ty] [tz] [rx] [ry] [rz]
```

where

- `[Posetag]`: Tag sent at the beginning of each pose message. This tag can be customized in the parameter file (see chapter 4).
- `[index]` : Integer uniquely describing the frame
- `[variance]`: Variance of the pose
- `[tx],[ty],[tz]`: Position of the object relative to the camera (in mm)
- `[rx],[ry],[rz]`: Orientation of the object relative to the frontal view (in rad)

See Chapter 3 for more details on the position and orientation format.

Nods format

The pose information sent via network has the following format:

```
[NodsTag] [Index] [State] [LogNod] [LogShake]
```

where

- `[Nodstag]`: Tag sent at the beginning of each Nods message. This tag can be customized in the parameter file (see chapter 4).
- `[index]` : Integer uniquely describing the frame
- `[State]`: State of the head nods and head shakes detector. Three possible states:
  - 0 : No head nods of head shakes detected
  - `1` : A head nod has been detected
  - `-1` : A head shake has been detected
- `[LogNod]`: Log likelihood of the HMM trained to detect head nods.
- `[LogShake]`: Log likelihood of the HMM trained to detect head shakes.

Server Protocol

Stereo images transfer

It is possible with Watson to grab stereo images on a remote system and send the images via network. Each stereo images received by Watson will be automatically processed when the transfer is completed.

Frame Header

Each stereo image sent must have the following header (ASCII standard):

```
F [FrameTag] [FrameIndex] [Focal] [CX] [CY]
```

where

- `[FrameTag]` : describe the type of information following the header. Each item following the header is represented by one capital letter. The order of each letters is not important. Here are the different items possible:
  - `I` : Intensity image for the referential camera
  - `Z` : Depth image for the referential camera
  - `R` : Intensity image of the referential camera
  - `O` : Region of interest of the tracked object

    - `P` : Pose of the object relative to the camera
- `[FrameIndex]` : Integer uniquely describing the frame
- `[Focal]` : Focal length of the referential camera (in pixel)
- `[CX]` : Center of the image along the X axis (in pixel)
- `[CY]` : Center of the image along the Y axis (in pixel)

After the header, each item described `[FrameTag]` must be sent via the network. The order they are sent is not important but the frame will not be processed until all items are received.

Image format

Each image sent have the following format (ASCII standard):

```
[ImageType] [Width] [Height] [BufferSize]
```

where

- `[ImageType]`: describe the type of image. Here the different items possible:
    - `I` : Intensity image for the referential camera;
    - `IC` : Compressed intensity image for the referential camera (JPEG);
    - `Z` : Depth image for the referential camera;
    - `ZC` : Compressed depth image for the referential camera (ZIP);
- `[Width]` : Width of the image
- `[Height]` : Height of the image
- `[BufferSize]` : Size of the (compressed, if specified) buffer (in byte)

Region of interest format

Each region of interest sent have the following format (ASCII standard):

```
ROI [offsetX] [offsetY] [width] [height] [nearZ] [farZ]
```

where

- `[offsetX]`: Horizontal offset of the region of interest
- `[offsetY]`: Vertical offset of the region of interest
- `[width]` : Width of the region of interest
- `[height]` : Height of the region of interest
- `[nearZ]` : Near boundary of the region of interest along the Z axis

- `[farZ]` : Far boundary of the region of interest along the Z axis

Pose format

Each pose sent have the following format (ASCII standard):

```
POSE [variance] [tx] [ty] [tz] [rx] [ry] [rz]
```

where

- `[variance]`: Variance of the pose
- `[tx],[ty],[tz]`: Position of the object relative to the camera (in mm)
- `[rx],[ry],[rz]`: Orientation of the object relative to the frontal view (in rad)

See Chapter 3 for more details on the position and orientation format.

Chapter 6

Programming Interface

Watson offers a C++ interface for the head tracking library and the head gesture recognition library. Using this interface, Watson can be used with different type of stereo cameras.

Sample programs

Watson comes with three sample programs:

- **SimpleSocket:** Shows how to connect to Watson via TCP/IP or UDP and how to receive tracking results. The TCP/IP example also shows how to start/stop Watson demo program remotely;
- **SimpleWatson:** This program shows how to grab images, track the head and detect head gestures using Watson DLL interface;
- **WatsonFromFile:** This program shows how to use Watson DLL interface to read intensity and depth images from disk, insert them into Watson grabbing sequence and track the head pose. This example can be extended to read images from a custom stereo camera.

All three samples program can be found in the directory \Watson\samples\. To compile them, you will need Microsoft Visual C++ .NET 2003 (msvc 7.1). For SimpleWatson and WatsonFromFile, the working directory should be set to ..\..\Sequences\Exemple.

Software architecture

Watson comes with two dynamics libraries:

- **Watson.dll:** This dynamic library contains all the functions related to grabbing and tracking. This is the main library for interfacing with the 3D object tracker. The internal structure of this library is described in the following subsections.

- **NodsShakes.dll:** This dynamic library contains specific functions for head nods and head shakes detection. Tracking results from Watson can directly be used in this library.

C++ Classes Overview

Main classes

The following classes are the main classes needed to interact with Watson:

Name	Inherit from	Description
CWatson		Main interface for the head pose tracker.
CNodsShakes		Main interface for the head gesture recognizer.
CSequence	list<CFrame>	Sequence of stereo images.
CFrame	CIPLImage3D	Stereo image with associated pose, velocity and 3D mesh.
CIPLImage3D		Stereo image with mask and region of interest (ROI).
Transformation		Rigid transformation (Rotation + Translation).
CIPLROI3D	vipiRoi	3D region of interest.
vipImage		Generic 2D image class (see following section)
vipiRoi		2D region of interest
CMesh		3D mesh.
CFaceMatch		Results from the face detector.

vipImage Image Library

This library implements a generic image wrapper for different color mode and storage types. It is based on the Image processing module of Intel Integrated Performance Primitives (IPP) library. The complete library of vipImage can be downloaded on SourceForge.net.

Name	Type	Typical use
vipImage8uC1	unsigned char	Grayscale images, mask images.
vipImage8uC3	unsigned char	Color images
vipImage8uC4	unsigned char	Color images with extra space for Alpha channel
vipImage16sC1	unsigned short	Disparity image

vipImage32fC1	Float	Depth image, X coordinates and Y coordinates.

Parameter Classes

The following classes contain the thresholds and parameters needed to track and recognize head gestures:

Name	Associated class	Description
CParamWatson	CWatson	High level parameters for the head pose tracker.
CParamNodsShakes	CNodsShakes	Parameters for the head gesture recognizer.
CParamSeq	CGrabSequence	Grabbing parameters for the stereo camera and model of the head (ROI).
CParamDirector	CTrackerDirector	Parameters for the online selection of keyframes and merging of the tracking results.
CParamInit	CInitTracker	Initialization criteria for the head tracker.
CParamMap	CMapBuilder	Parameters for the insertion of new keyframes (view-based appearance model).
CParam3DModel	C3DModel	Parameters for the ellipsoid matching algorithm.
CParamTrackerICP	CTrackerICP	Parameters for the core differential tracker.
CRecordParam	CGrabSequence	Record parameters for saving offline sequences.
CParamSimple	CTrackerSimple	Parameters for the face detection and segmentation.

Detailed Interface

The following subsections list and describe the member functions of the most important classes.

CWatson

Grabbing images

- **GrabNewFrame():** Utility function that automatically calls AcquireImages, GetImages and InsertImages.
- **AcquireImages():** Acquires the images from internal Grabber.
- **GetImages():** Returns images acquired by the internal Grabber.
- **InsertImages():** Crops the image (if necessary), compute ROI, compute depth and insert frame inside the ImageSequence (calls InsertFrame).

- **InsertFrame():** Inserts frame (intensity and depth) inside the ImageSequence.

Stereo images can be grabbed automatically using the internal Grabber or inserted manually using one of the Insert function. If you decide to use the internal Grabber (which supports VidereDesign cameras and pre-recorded sequences), you should use the utility function **GrabNewFrame()**. The functions **AcquireImages()**, **GetImages()** and **InsertImages()** can be used if you want to multi-thread the processes of grabbing and stereo. If you decide to insert manually your images (i.e. because you are using a different camera/stereo algorithm), you should use **InsertImages()** or **InsertFrame()**. **InsertImages()** takes as input the left and right images and compute the stereo internally. To work properly, you will need a valid license of Small Vision System (SVS). If you already computed the stereo, then you should use **InsertFrame()** to insert the depth image with its associated intensity image.

Tracking

- **ProcessNewFrame():** Segments face, detects face (if activated) and tracks head.
- **SetMode():** Set tracking state of Watson (see description below).
- **SetAutoDetection():** Activates the face detection for automatic initialization of the head tracker.
- **SetAutoReinit():** Set if the tracker should automatically reinitialize when the user move too fast or not enough valid pixels are present.
- **SetRoi():**

Results

- **GetCurrentFrame():** Returns current frame (with associated pose and velocity).
- **GetFrameSeq():**
- **GetLinkList():**
- **Reset():** Cleans the image sequence and the model (if autoClean == true), and resets the tracker.

Keyframes

- **CleanMap():** Erases all the keyframes from the view-based appearance model.
- **SetAutoClean():** Set if the view-based appearance model should be erased every time the tracker is reinitialized.
- **GetMapSeq():**
- **LoadMap():**
- **SaveMap():**

Recording

- **StartRecording()**
- **StopRecording()**
- **LoadSequence()**
- **SaveSequence()**
- **ReloadSequence()**

Face detector

- **GetNbFaceMatches()**
- **GetListFaceMatches()**
- **GetCommonMask()**
- **DrawBoxes**
- **getCountDown()**

Parameters

- **GetParamWatson()**
- **GetParamInit()**

- **GetParamSimpleTracker()**
- **GetParamRecorder()**
- **GetParamMap()**
- **GetParamDirector()**
- **GetParamICP()**

CFrame

Images

- **GetIntensityImage()**
- **GetIntensityRightImage()**
- **GetColorImage()**
- **GetDepthImage()**
- **GetXImage()**
- **GetYImage()**
- **GetMask()**
- **GetValidDepth()**

Calibration

- **BackProject()**
- **GetFocalLength()**
- **GetImageCenterX()**
- **GetImageCenterY()**
- **GetDeltaX()**
- **GetDeltaY()**

Pose

- **GetPose()**
- **GetVelocity()**
- **GetRoi3D()**
- **GetCenterX()**
- **GetCenterY()**
- **GetCenterZ()**

Pose

- **GetFrameIndex()**
- **GetTimeStamp()**
- **isKeyframe()**

Transformation

- **GetEulerAngle()**
- **GetRotationMatrix()**
- **GetTranslation()**
- **GetPtrTransformationMatrix()**
- **GetVariance()**
- **ApplyTransform()**

CNodsShakes

Detection

- **InsertLink()**
- **Reset()**

- **Enable()**
- **IsEnabled()**

Results

- **GetCurrentState()**
- **GetLLNods()**
- **GetLLShakes()**
- **GetCurrentTimeStamp()**

Chapter 7

Troubleshooting

In this chapter, we describe solutions to common problems/mistakes happening when installing Watson.

"Can't open frame grabber"

Problem

When starting Watson, a message saying "Can't open frame grabber" is displayed in the DOS prompt and no intensity image (F5) or depth image (F6).

Solution

This error message usually signifies that the stereo camera has not been installed properly.

- Check if the stereo camera is connected☺. You should be able to see a red light from the front of the stereo camera.
- For Videre Design cameras, SVS must be installed before plugging the camera. If you have a Mega-D stereo camera, check the Device manager to be sure that the camera is recognized as a PixelLink™ imaging module.
- Be sure that you are using the appropriate svsgrab.dll file. If you have a Mega-D you should run setup_megad.bat and if you have a DCS, you should execute the file setup_dcs.bat.

"Can't start continuous capture"

Problem

When starting Watson, a message saying "Can't start continuous capture" is displayed in the DOS prompt and no intensity image (F5) or depth image (F6).

Solution

This error message usually signifies that the camera is not responding.

- Unplug and replug the camera. When a program stop during the grabbing process, the camera must be reset.

Bad stereo images or blank stereo image

Problem

The stereo images (F6) looks noisy (or you get a blank image) but you get an intensity image (F5).

Solution

This happens usually if you are using the wrong calibration.

- If you change the lens on your stereo camera or if you get a new camera, you should always recalibrate the stereo camera. Please refer to SVS documentation for more information on how to calibrate your camera.
- When calibrating the camera, be sure to use SVS42d.exe. Some older versions of SVS may also work.
- Be sure that you modified the parameters file ParamSeqDirect.cfg so it uses your new calibration file. To do so, open the file, search the field CONFIG_FILENAME: and modify its value to be the name of your calibration file (calib-xxxxx.ini).

Tracker doesn't initialize

Problem

The images are grabbed properly but the head tracker never starts.

Solution

When in Auto-init mode, the head pose tracker initialize after it detected a face.

- Check that the auto-initialization is turned on. In the demo program, you can press CTRL+A to toggle the auto-init option. In the parameter file, you can set the option AUTO_INIT: TRUE.
- The Adaboost-based face detector uses parameter files placed in the directory \Watson\Classifier. If you receive the error message "Cannot open file classifier.txt to read." During the startup, this means that Watson could not find these files.
- The face detector checks for faces at different scales. You can increase the parameter NUMBER_SCALE: 4 to a larger value so that closer faces are detected.
- When a face is detected, Watson checks that the face is inside a certain depth range. You can modify this range of valid detection using the parameters MIN_DEPTH_MASK and MAX_DEPTH_MASK.

- Finally, Watson will initialize only after a face has been detected for a certain time. You can reduce the number of frame detected using the parameter NB_DETECT_BEFORE_INIT.

Bad head tracking results

Problem

The head is detected but doesn't seem to be tracked properly.

Solution

- Try to increase the gain of the camera. Sometime when the images are too dark, the intensity gradient computed during the tracking become too noisy. Also, some internal parameters for key-frame selection depend on the intensity of the image.
- Be sure that you are using the right calibration file. Watson comes with a default calibration file (calib.ini) which should be replaced by the appropriate calibration file that you created using SVS. The quality of the tracking will improve dramatically if you use the right calibration file for your camera.

Watson crashes during grabbing/tracking

Problem

Watson crashes sometime on Pentium 4 HT.

Solution

- Turn off the hyper-thread option in your BIOS.

References

[1] Louis-Philippe Morency and Trevor Darrell, From Conversational Tooltips to Grounded Discourse: Head Pose Tracking in Interactive Dialog Systems, International Conference on Multimedia Interfaces, College State, PA, 2004

[2] Morency, L.-P., Rahimi, A. and Trevor Darrell, Adaptive View-based Appearance Model, Proceedings of IEEE conference on Computer Vision and Pattern Recognition, 2003

[3] Morency, L.-P., and Trevor Darrell, Stereo Tracking using ICP and Normal Flow, Proceedings of International Conference on Pattern Recognition, 2002

[4] http://www.euclideanspace.com/maths/geometry/rotations/conversions/angleToMatrix/index.htm