PrimeSense

PrimeSense ™

NITE Controls User Guide

March, 2011

PrimeSense Confidential

COPYRIGHT © 2011 PrimeSense™ Inc.

ALL RIGHTS RESERVED. NO PART OF THIS DOCUMENT MAY REPRODUCED, PHOTOCOPIED, STORED ON A RETRIEVAL SYSTEM OR TRANSMITTED WITHOUT THE EXPRESS PRIOR WRITTEN PERMISSION OF PRIMESENSE IN

About This Document

This document provides detailed description of PrimeSense's NITE Controls middleware for software developers.

PrimeSense Confidential

Table of Contents

Table of Figures

PrimeSense Confidential

PrimeSense Confidential

Disclaimer and Proprietary Information Notice

The information contained in this document is subject to change without notice and does not represent a commitment by of PrimeSense, Inc. PrimeSense, Inc. and its subsidiaries make no warranty of any kind with regard to this material, including, but not limited to implied warranties of merchantability and fitness for a particular purpose whether arising out of law, custom, conduct or otherwise. While the information contained herein is assumed to be accurate, PrimeSense, Inc. assumes no responsibility for any errors or omissions contained herein, and assumes no liability for special, direct, indirect or consequential damage, losses, costs, charges, claims, demands, fees or expenses, of any nature or kind, which are incurred in connection with the furnishing, performance or use of this material. This document contains proprietary information, which is protected by U.S. and international copyright laws. All rights reserved. No part of this document may be reproduced, photocopied or translated into another language without the prior written consent of PrimeSense, Inc.

PrimeSense Confidential

1 Overview

1.1 Natural Interaction by PrimeSense

1.1.1 Natural Interaction

The term Natural Interaction (NI) refers to a concept whereby Human-device interaction is based on human senses, mostly focused on hearing and vision. Human device NI paradigms render external peripherals, such as remote controls, keypads or a mouse obsolete. Examples of everyday NI usage include:

- Speech and command recognition, where devices receive instructions via vocal commands.
- Hand gestures, where pre-defined hand gestures are recognized and interpreted to activate and control devices. For example, hand gesture control enables users to manage living room consumer electronics with their bare hands.
- Body Motion Tracking, where full body motion is tracked, analyzed and interpreted for gaming purposes.

1.1.2 3D Sensing by PrimeSense

The PrimeSense 3D Sensor is a device that allows computers to perceive the world in 3D and build an understanding of this world, just as humans do. The device includes a sensor component that observes a scene (users and their surroundings), and outputs a depth image of the scene. A depth image is an image in which each pixel represents its distance from the sensing device. However, the depth images produced by the PrimeSensor are raw data. They need to be streamed to a processing component that can understand it the way humans that observe the real environment do. In other words, the stream of depth images needs to be translated to meaningful information. This information can specify, for example, where the people in the scene are, which object is identified as furniture, where the walls and the floor are, and which movements are performed by the people. This component is the 'brain' that can see the scene, analyze it, track the humans within the viewed area and react to their movements.

1.2 What is NITE?

PrimeSense's Natural Interaction Technology for End-user - NITE™ - is the middleware that perceives the world in 3D, based on the PrimeSensor depth images, and translates these perceptions into meaningful data in the same way that people do. The PrimeSensor™ 3D sensor observes the users' environment, while NITE acts as the perception engine that comprehends the user interaction within these surroundings. NITE middleware includes both computer vision algorithms that enable identifying users and tracking their movements, as well as framework API for implementing Natural-Interaction UI controls that are based on user gestures.

PrimeSense Confidential

1.2.1 NITE Control Paradigms

The concept on which NITE middleware was designed is based on two key paradigms: the hand control paradigm and full body control paradigm.

Hand Control occurs when users control an application using hand gestures. This is the common case when talking about a living room experience, such as controlling the TV, controlling content menus of media centers and more.

NITE Hand-based control paradigm distinguishes between two modes:

1. Focus gesture detection mode – during this time, the system observes the scene to detect a request for control in the form of a focus gesture, such as a hand wave performed by one of the users in the scene. This mode is typically useful, for example, when the system is in passive mode, and there is no user currently controlling it.
2. Control mode – when a user requests and gains control, the system tracks the controlling hand(s), looking for specific gestures.

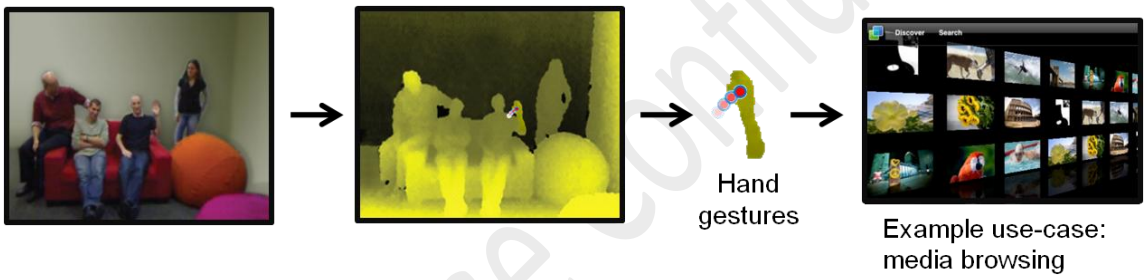

Figure 1: Hand Based control

Hand-based control is illustrated in Figure 1. The image shows a living room crowded with viewers. NITE is able to track the palms of all the viewers, creating a pool of viewers who are candidates for requesting to gain control over the TV. Yet only one viewer is controlling the TV at any given time – the user who performed the Focus gesture. NITE is able to identify this viewer and prevent the other viewers from interfering with the on-going interaction. The illustration shows NITE locked on a specific viewer, tracking the viewer's palm and distilling hand gesture widgets, enabling the viewer to control a media browsing application. A gesture widget is a predefined palm movement, e.g. a wave, a circle, a push, a left-right slider, an up-down slider, etc. NITE tracks the palms until identifying a complete widget maneuver, at which time it issues a call to the application with the identified gesture widget. The application simply needs to assign an input control per each defined widget.

Full Body natural interaction is commonly associated with gaming, where a player(s) stands in front of the TV having an immersed gaming experience. The goal of NITE's full-body-based control module is to extract the important skeleton features and track those through time. As a result, the application is not exposed to the depth data, but rather it receives simple point skeleton and skeleton gestures to which it can easily assign game control inputs.

PrimeSense Confidential

The goal of the NITE is not to overwhelm the host with an infinite amount of depth data, but rather enable computationally inexpensive extraction of user intentions. This is illustrated in Figure 2: while processing a 30fps depth stream is not trivial, the required end-result is merely several joint points outlining the player's full-body skeleton, and tracking of these points throughout every frame.

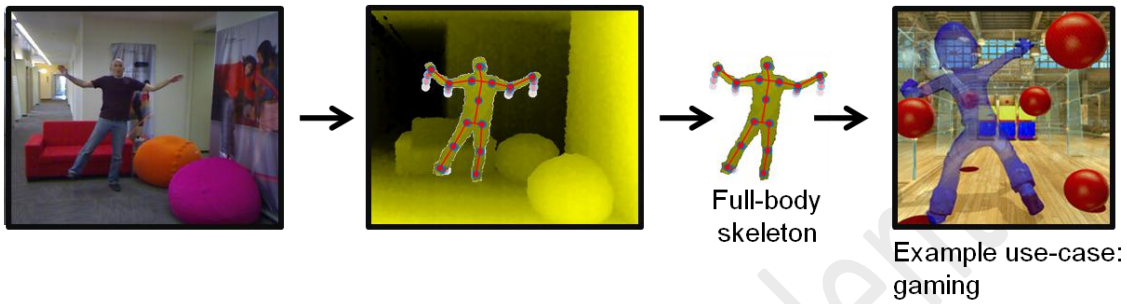

Figure 2: Full Body Control

1.2.2 NITE Algorithms and NITE Controls

As mentioned earlier, NITE middleware includes both a computer vision engine, and a framework that provides the application with gesture-based UI controls. This division is essentially the two layers that compose NITE:

1. **NITE Algorithms** - The lower layer that performs the groundwork of processing the stream of raw depth images. This layer utilizes computer vision algorithms to perform:
    a. Scene segmentation – a process in which individual users and objects are separated from the background and tagged accordingly.
    b. Hand point detection and tracking - Users hands are detected and followed.
    c. Full body tracking – Based on the Scene segmentation output, users' bodies are tracked to output the current user pose - a set of locations of body joints.

    NITE Algorithms is an OpenNI compliant module (see section 1.2.2.1).
2. **NITE Controls** - A layer of application framework that analyzes the hand points generated by NITE Algorithms, and provides tools that enable application writers to design the flow of the application according to the hand movements. NITE Controls framework identifies specific gestures, and provides a set of UI controls that are based on these gestures.

PrimeSense Confidential

1.2.2.1 What is OpenNI?

The OpenNI organization is an industry-led, non-profit organization, formed to certify and promote the compatibility and interoperability of Natural Interaction (NI) devices, applications and middleware.

As a first step towards this goal, the organization has made an open source multi-language, cross-platform framework available – the OpenNI framework – which provides an application programming interface (API) for writing applications utilizing natural interaction.
The main purpose of OpenNI is to form a standard API that enables communication with both:

- Vision and audio sensors (the devices that 'see' and 'hear' the figures and their surroundings.)
- Vision and audio perception middleware (the software components that analyze the audio and visual data that is recorded from the scene, and comprehend it). For example, software that receives visual data, such as an image, returns the location of the palm of a hand detected within the image.

OpenNI supplies a set of APIs to be implemented by the sensor devices, and a set of APIs to be implemented by the middleware components.
OpenNI is an open source API that is publicly available at www.OpenNI.org

PrimeSense Confidential

1.2.3 Abstract Layered View

Figure 3 below displays a layered view of producing, acquiring and processing depth data, up to the level of the application that utilizes it to form a natural- interaction based module.

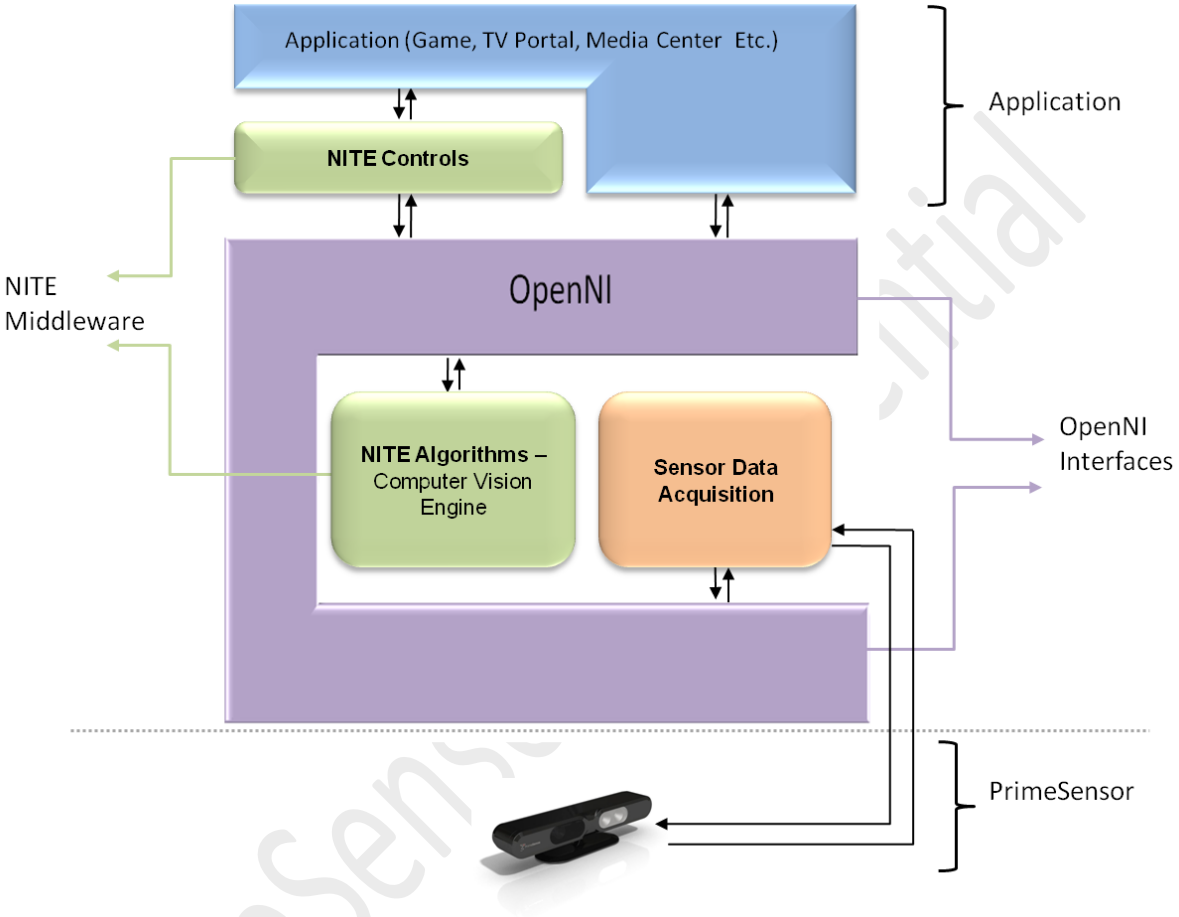

Figure 3: Layered View of Depth Acquiring and Processing

- The lower layer is the PrimeSensor device, which is the physical acquisition layer, resulting in raw sensory data – a stream of depth images.
- The next layer, which is C-shaped (already executed on the host – a PC, a Set Top Box, etc.) represents OpenNI. OprnNI provides communication interfaces that interact with both the sensor's driver and the middleware components, which analyze the data from the sensor.
- The Sensor data acquisition is a simple acquisition API, enabling the host to operate the sensor. This module is OpenNI compliant – the interfaces it exposes conform to OpenNI API standard. In other words, this is an OpenNI plug in.
- The NITE Algorithms layer is the computer vision middleware and is also plugged into OpenNI. It processes the depth images produced by the PrimeSensor.
- The NITE Controls layer is an applicative layer that provides application framework for gesture identification and gesture-based UI controls, on top of

the data that was processed by NITE Algorithms. NITE Controls communicates with NITE Algorithms via the standard interfaces and data types defined by OpenNI.

- The top-most layer is the natural-interaction-based application. This application can use NITE Controls, and can also approach OpenNI directly in order to access the data generated by NITE algorithms, or even the raw data produced by the sensor.

PrimeSense Confidential

2 NITE Controls

2.1 Session Management

For an application that uses NITE Controls, a session is a state in which the user is in control of the system using his or her hand. During the session, the hand point is tracked by the system, and has persistent ID. A session typically starts with the user requesting to gain control, by performing a pre-defined hand gesture (for example, waving). This hand gesture is referred to as the '**focus gesture**' – this is the gesture that once identified, the system interprets as a request from the user to gain the control. As of this point, the user's hand is tracked. The session ends when the user's hand disappears.

2.1.1 Session States

There are 3 possible session states:

1. Not in Session: In this state, there is no active session, hence the system is in a mode of scanning the scene to detect a focus gesture. Once this gesture is recognized, the state changes to "in session".
2. In Session: In this state, there is a hand that is currently in control and being tracked by the system.
3. Quick Refocus: This is an intermediate state, in which, while in session, the hand point is lost. We don't want to stop the session yet, but rather give a grace period in which the session can be resumed with a different (perhaps shorter or easier) hand gesture, which we will refer to as the 'quick refocus gesture'. While in "Quick Refocus" state, the session can also be resumed using the regular focus gesture. Once the grace period has timed out, the state changes to "Not in Session". The "Quick Refocus" state is optional.

Figure 4 illustrates the session states flow.

PrimeSense Confidential

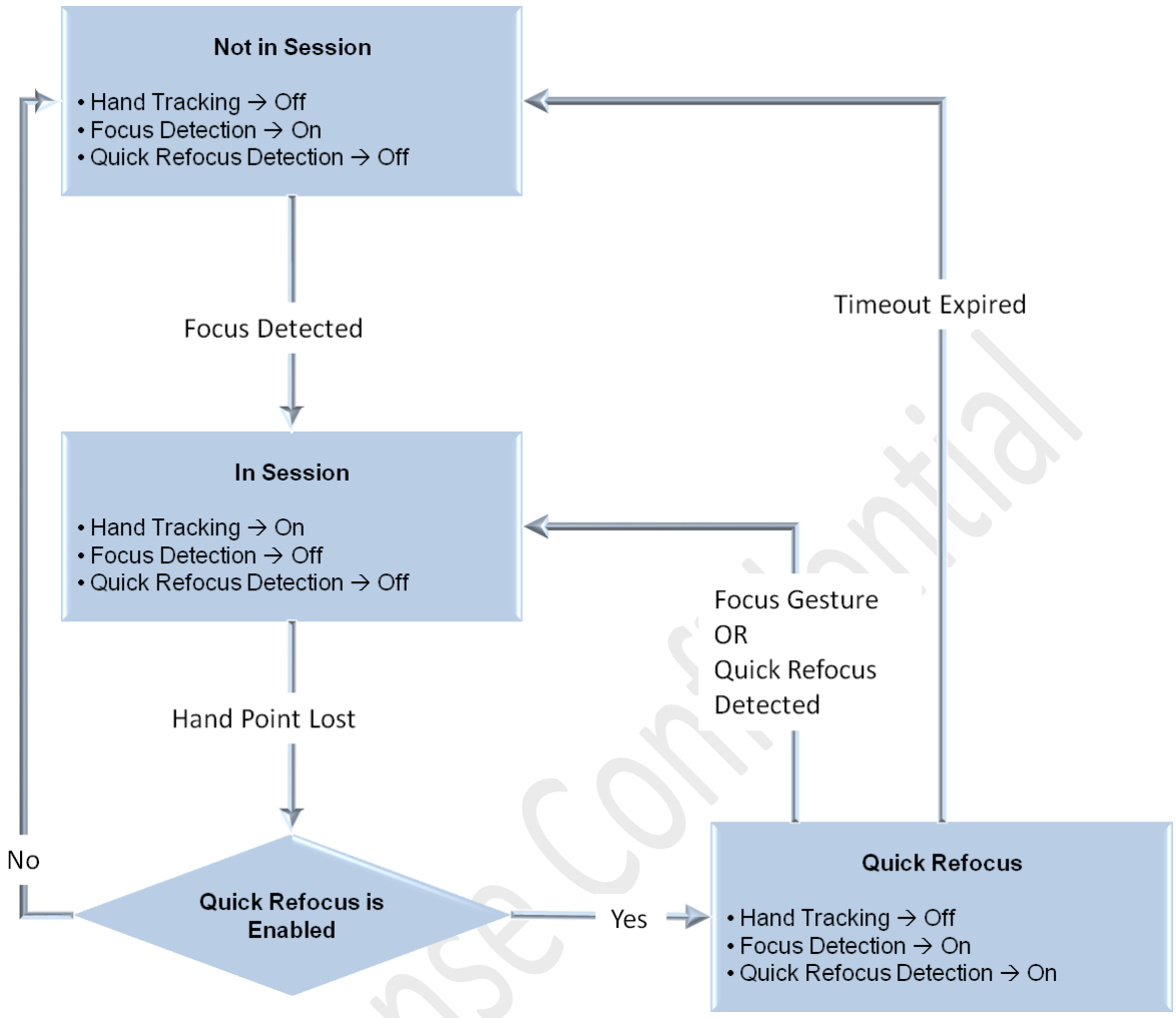

Figure 4: Session States Flow

The preliminary state is 'not in session', where the system scans the scene to detect a focus gesture. When a focus gesture is identified, the state changes to 'In Session', and the hand that performed the focus gesture is tracked. When the tracked hand is lost, the state changes to either:

- Not in Session – if the 'Quick Refocus' state is disabled. In this state, the system again expects a focus gesture.
- Quick Refocus - if enabled. In this state, the system expects the focus gesture or the quick refocus gesture.

If any of them is identified before a certain configurable timeout is reached, the state changes to 'In Session'. If none of them is identified before the timeout expires, the state changes to 'Not in Session'.

PrimeSense Confidential

2.2 Point Controls

NITE Point controls are a set of objects that, for each frame, receive a stream of the current active hand points from some source (for example, the session manager), analyze it and perform some action. The Point controls attempt to reach a meaningful conclusion from the hand points' behavior. In the current implementation, all the point controls operate on the primary point only, apart from the Steady detector.

2.2.1 Primary Point

All of NITE hand-point controls are fed with points that relate to one specific hand that is currently defined as the active hand, or the primary point. The primary point is the first recognized hand point (possibly the one that performed the focus gesture). If the primary point is no longer available (fails to be identified), and other points exist, one of these points will take over as the primary point. The primary point is set by the session manager.

2.2.2 Messages

The flow of data between NITE controls is done through messages. Messages can be sent by Message Generators and be listened to by Message Listeners.
The base object of all the point controls is a Message Listener. There are different types of messages, e.g. PointMessage, ActivationMessage, SessionMessage etc. Any Listener that receives a message that is intended for use, must make sure that it is indeed of the expected type. Once the type is verified, the control handles that message. In most cases, it extracts the actual data from the message, and either forwards it for handling by a subclass, or analyzes it and supplies the subclass (or user callbacks) with specific events.

For instance, an XnVPointControl receives an XnVPointMessage. It extracts an XnVMultipleHands object from it, which is a container of all the active hand points currently identified. It also analyzes it to find which points are new (in order to dispatch the OnPointCreate callback), which points already exist (resulting with OnPointUpdate callback), and which points that were previously tracked are no longer identified (the OnPointDestroy callback). It also finds out which point is the primary point, and continues to call the relevant primary point callbacks.

2.2.3 Point Control Events

Each point control supports a set of events to which callbacks can be registered. For every frame, all the events that occurred in the frame are dispatched. For example, the event of 'No Points' will be dispatched for each frame where no hand points exist in the scene.
The following events can be registered to from any of the point controls:

1. New point was created
2. Existing point moved

PrimeSense Confidential

3. Point that existed disappeared
4. Primary point was created
5. Primary point was moved
6. Primary point replaced – the primary point is no longer available, but another hand point exists and has taken over as the primary point.
7. Primary point destroyed – this in fact means that there are no more points (hence the primary point cannot be replaced).
8. No Points – this event is dispatched when there are no points at all.

In addition, each control provides a set of specific events that relate to the control's functionality.

2.2.3.1 Push Detector

This point control tries to recognize a push gesture performed by the tracked hand point. A push is detected when a certain velocity is reached in an angle close to the Z-axis for a certain period of time.
This gesture is typically used for selecting items.
The following parameters can be configured:

- Immediate Duration – the period of time in which a movement is detected as push
- Immediate Minimum Velocity – the minimum velocity in the time span to define as push.
- Immediate Offset –
- Maximum angle between immediate and Z
- Minimum angle between immediate and previous
- Previous duration
- Previous minimum velocity
- Previous offset

Supported events:

- Push - called when complete push gesture is detected
- Stabilized – The stabilized event happens after a push is detected, when the hand is relatively static. Another push is not allowed until a stabilized event has been detected.

2.2.3.2 Swipe Detector

This point control attempts to recognize hand point movement as a swipe motion, either up, down, left or right. A swipe motion is a short movement in a specific direction, followed by the hand resting.
The following parameters can be configured:

- Motion Speed Threshold – the minimal speed for recognizing a gesture as a swipe.

PrimeSense Confidential

- Motion Time - The minimal duration for recognizing a gesture as a swipe.
- X Angle Threshold – the allowed deviation from X-axis, to recognize a gesture as a Swipe left or right.
- Y Angle Threshold – the allowed deviation from Y-axis, to recognize a gesture as a Swipe up or down.

Supported events:

- Swipe up
- Swipe Down
- Swipe Left
- Swipe Right
- Swipe – a general swipe event that is called when a swipe at any direction is detected.

2.2.3.3 Steady detector

This point control tries to detect when a hand point has been steady for some time. Steady means that the user's intention is to not move the hand, while the detection takes into account that a person's hand may still slightly move when held steady. The steady detector uses the variance of the hand point within a specific time frame. Steady is mostly useful for the detection of a resting hand between gestures – to clearly signal that one gesture has ended and another one is (potentially) about to start.
The following parameters can be configured:

- Detection duration - the minimum time that should pass where the hand is steady in order to conclude a steady state. Default is 200 milliseconds.
- Standard deviation – the deviation of the hand point that can be considered as steady.

Supported events:

- Steady hand detected
- Not Steady - After a hand is declared as steady, its next movement triggers the Not Steady event

2.2.3.4 Wave detector

This point control attempts to recognize hand point movement as a wave gesture. A wave is a number of direction changes within a timeout. By default, four direction changes are needed to identify a wave.
The following parameters can be configured:

- Flip Count – the number of direction changes that constitute a wave gesture
- Max Deviation –the allowed deviation from the motion plane, since the motion is expected to be right-left.
- Min length – the length (in mm) of the motion before a direction change (a flip).

Supported events:

- Wave – called when a complete wave gesture was detected.

PrimeSense Confidential

2.2.3.5 Circle Detector

This point control attempts to recognize hand point movement as a circular motion. The Circle Detector needs a full circle in either direction (clockwise or anti-clockwise) in order to acknowledge the movement as a circle.
The following parameters can be configured:

- Minimum Points – the minimal number of points to try to match as a circle.
- Min/Max Radius – The minimal/maximal radius for which a circle is valid
- Close to Existing Circle – how close may the new point be to the existing circles circumference to be considered the same circle
- Existing Weight – The circle size and center is always changing according to the hand points that are added. Each new point is taken into consideration for the description of the circle (its location and radius). This parameter controls the weight that each object gets – the existing circle compared to the new point - when coming to decide how to change the circle description.
- Max Errors – determines the number of allowed errors, where errors are points that do not match the circle. When the maximal number of points is reached, the circle is no longer valid.

Supported events:

- Circle
- No Circle – It is possible to decide at any point that an existing circle no longer matches a circle. This may be because there are too many errors in matching to a circle, or because the hand point is no longer available, or because the user requested the circle to be ignored.
-

2.2.3.6 SelectableSlider1D

This point control attempts to recognize hand point movements along a slider, aligned to any of the 3 axes - X (left-right), Y (up-down) or Z (close-far). The control provides a value between 0 and 1, per frame, indicating where the hand point is relative to the preset ends of the slider.
In addition to identifying motion along the sliding axis, it is possible to define selectable areas within the slider. This is done by dividing the sliding zone equally into a number of areas, where each area defines a single selectable item in the slider. It can be used, for example, to implement menus, with each item being a single menu option.
The following parameters can be configured:

- Item Count – the number of items to create in the slider
- Border Width – it is possible to define a border for the slider – an area in the slider edges that belongs to the slider, but is not selectable. When the hand point hovers over this area, the ‘scroll’ event is dispatched.
- Hysteresis Ratio – this parameter defines some margin for dispatching an event for entry to an item. The margin defines how deep into the item a hand point should be, in order to change the focus to this item. The purpose of this is to

PrimeSense Confidential

avoid switching items when the hand point is close to the border between two items, due to the fact that the hand point may be trembling
- Slider Size – the real-world size of the slider in mms.

Supported events:
- Item Hover – when the hand point hovers over an item of the slider
- Item Selected – when a selection is made to an item of the slider (selection is done by movement in either of the 2 axes that are not the primary slider axis)
- Off Axis Movement – when movement is detected in a direction that is off-axis to the primary axis of the slider, but before the movement is considered selection.
- Scroll – called when the hand point is in the slider's border area – an area that belongs to the slider but is not selectable. 'Scroll' event will be called with a value of (-1,0) if the hand point is at the left/down/far border, or (0,1) if the hand point is detected at the right/up/near border.
- Value Change – indicates the location of the hand point inside the slider, relative to the slider edges.

2.2.3.7 SelectableSlider2D

This point control attempts to recognize hand point movement on a predefined X-Y plane. The control provides two values between 0 and 1, per frame, indicating where the hand point is in both X and Y axes, relative to the ends of the slider.
The SelectableSlider2D also enables dividing the plane into equal areas - each area defines a single selectable item in the plane. Selection on an item is done by movement along the Z axis (the "depth" axis).
It allows registration to events for when the hand is over a different item, when that item is selected (by movement along the Z axis), and for each frame, two values between 0 and 1, indicating where the hand point is in both axes, relative to the ends of the slider
The following parameters can be configured:
- Item Count – the number of items (represented by number of rows and columns) to create in the selectable area
- Item X Count – the number of rows to create in the selectable area
- Item Y Count – the number of columns to create in the selectable area
- Border Width – it is possible to define a border for the slider – an area in the slider edges which belongs to the slider, but is not selectable. When the hand point hovers over this area, the 'scroll' event is dispatched.
- Hysteresis Ratio – this parameter defines some margin for dispatching an event for entry to an item. The margin defines how deep into the item a hand point should be, in order to change the focus to this item. The purpose of this is to avoid switching items when the hand point is close to the border between two items, due to the fact that the hand point may be trembling.
- Slider Size – the real-world size of the slider in mms.

PrimeSense Confidential

Supported events:

- Item Hover – when the hand point hovers over an item of the slider
- Item Selected – when a selection is made to an item of the slider (selection is done by movement in Z axis)
- Off Axis Movement – when movement is detected in a direction that is off-axis to the primary axis of the slider, but before the movement is considered selection
- Scroll - called when the hand point is in the slider's border area – an area that belongs to the slider but is not selectable. 'Scroll' event will be called with a value of (-1,0) if the hand point is at the left/down/far border, or (0,1) if the hand point is detected at the right/up/near border.
- Value Change – indicates the location of the hand point inside the slider, relative to the slider edges

2.3 Flow Objects and NITE Tree

NITE Controls provide a framework that enables the application writer to define an applicative flow, which determines the flow of data between the controls, and defines conditions for when a control is active, that is, receiving data.
When NITE Controls are connected directly to a Session Manager, it implies that all the controls are always active and are receiving hand points whenever there are any. When a flow is defined, the controls receive the hand point data according to the current state of the state machine where the flow is defined.

2.3.1 NITE Tree

The flow of data (hand points) between NITE objects can be depicted using a graph, most commonly in the form of a tree.
This NITE tree describes the possible states for data flow, and eventually details the flow of data at any single point in time.

2.3.2 Flow Objects

Flow objects are objects that determine the flow of data between the controls.

2.3.2.1 Router

The flow router object sends all the data it receives to a single object that is connected to it. This object is referred to as the 'active' object - the only object that receives the data that is sent from the flow router at a given point in time. The active object may be replaced according to different states of the application flow. When an active object is replaced, it becomes inactive and will no longer receive data from the flow router.

Figure 5 displays the NITE tree for an application that is searching for a swipe gesture and a circle gesture.

PrimeSense Confidential

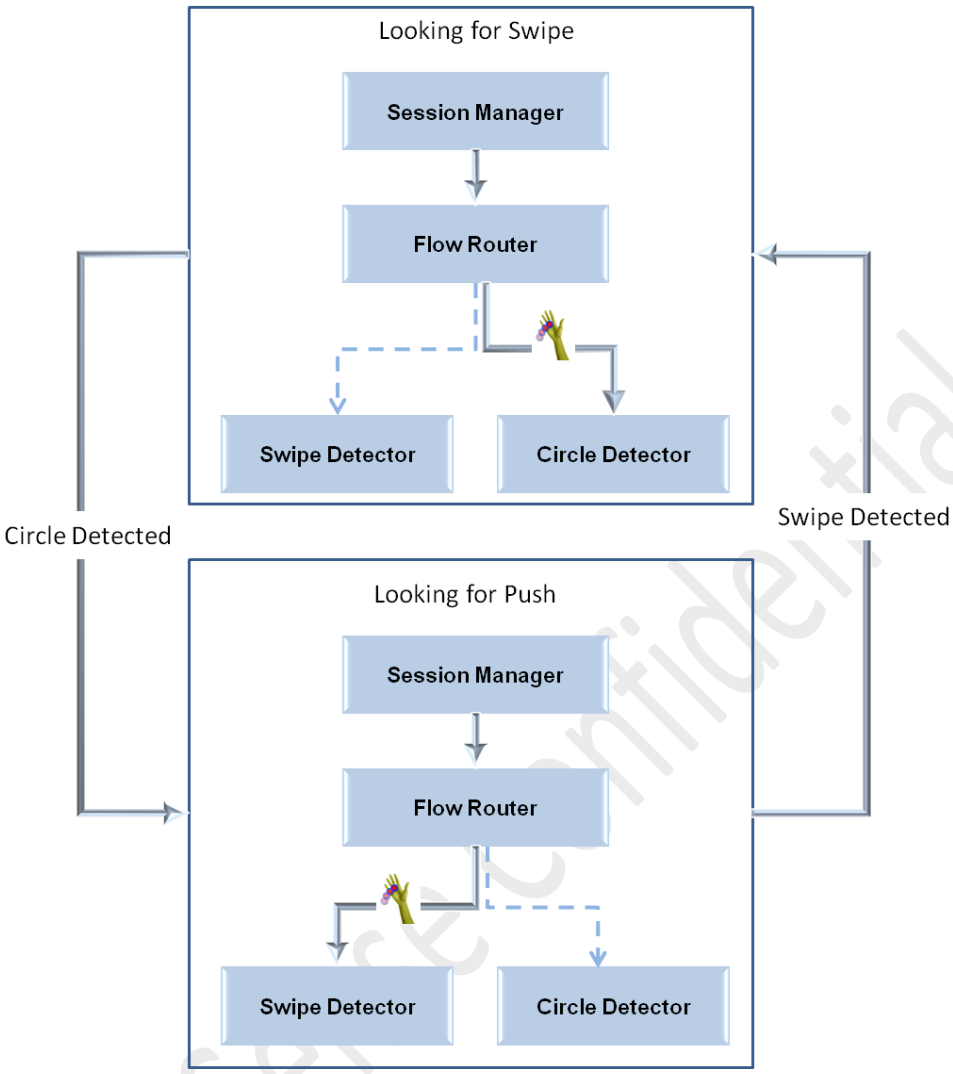

Figure 5: Flow Router Usage Example

The application switches between the active object (either swipe detector or circle detector) according to the applicative logic. For example, a swipe gesture starts a game, and a circle gesture ends it. So while the system is in standby, it looks for a 'start game' command in the form of a swipe gesture. Once a swipe is detected and the game begins, the circle detector becomes the active object, since the system is now looking for an 'end game' command in the form of a circle gesture.

2.3.2.2 Broadcaster

The broadcaster object sends all the data (hand points) it receives to all the objects that are connected to it.

Figure 6 illustrates a broadcaster object that sends hand data to three point controls at the same time: a push detector, a swipe detector and a circle detector.

PrimeSense Confidential

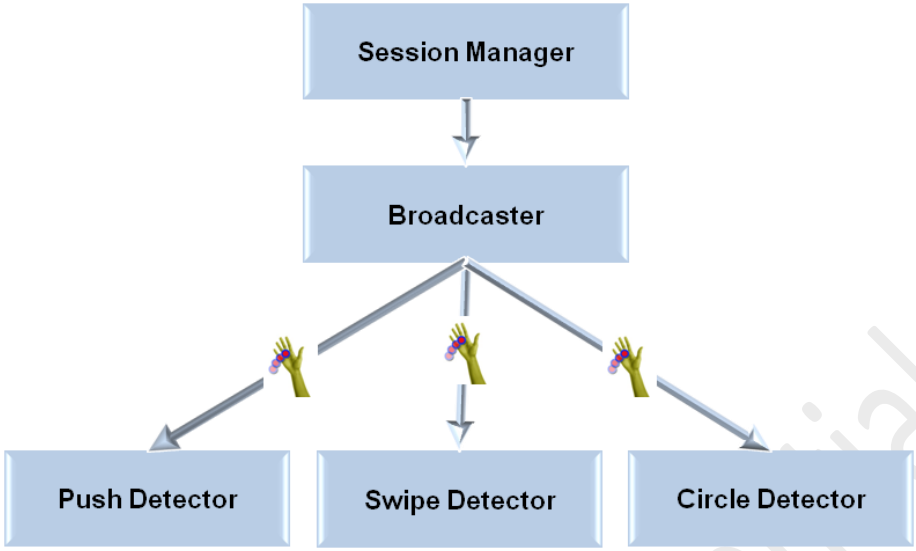

Figure 6: Broadcaster Usage Example

2.3.3 Filter Objects

Filters are objects that receive a stream of all the tracked hand points and perform some processing on the raw points data, according to specific logic. The output of the filter objects can include all points that have been processed and possibly altered, or only those points that comply with specific conditions.

2.3.3.1 Point Denoiser

The Point Denoiser executes a smoothing algorithm on all the hand points, in order to eliminate small movements that are most likely created by the user's hand not being entirely stable.

2.3.3.2 Point Area

The Point Area enables defining a 3D area and filtering out (or hiding) points that are out of this area. Hiding the points is referred to as 'silencing' them. The Point Area receives all the hand points, and passes on only points that are within the configured 3D area. There are events for when a point is silenced (moves outside of the bounding box), when a point is revived (was silenced but is now detected back inside the bounding box) and when a silenced point is destroyed.

Supported events:

- Point Silenced – Called when a point exits the 3D area
- Point Revived – Called when a point that previously exited the 3D area is now traced again inside the area
- Point Removed – Called when a point actually disappears from the scene.

PrimeSense Confidential

2.4 Creating Compound Controls

NITE Controls enables its users to compose new controls, based on the existing ones. In order to generate a new control, the first step is to define a NITE tree that represents the data flow between the controls that contain this new control. The new control listens to messages and forwards them to the embedded NITE tree, where they are handled according to the flow defined by the tree.

Example: Box Control

This example illustrates a new compound control called "box control", which is demonstrated in the "Boxes" sample of NITE. The box control is a control that searches for a push hand gesture, and in parallel, for either a swipe or a steady hand gesture.

- A swipe gesture highlights one of the Box's walls, according to the swipe direction. A swipe up highlights that upper wall; a swipe left colors the left wall etc.
- A steady detection is used to detect any resting between swipe movements, in order to start listening to swipe again.
- Detection of a push gesture at any time results in exiting the box control.

The advantage of implementing the box as a new compound control is that having three boxes is as easy as one – all that is required is the creation of additional instances of the same compound control.

The "Boxes" sample displays a slider with three box controls above it. The sample enables selecting one of the boxes using the slider. Once a box is selected, it becomes the active control – the control that currently listens to hand point messages.

Figure 7 illustrates the NITE tree that defines the data flow for the box control.

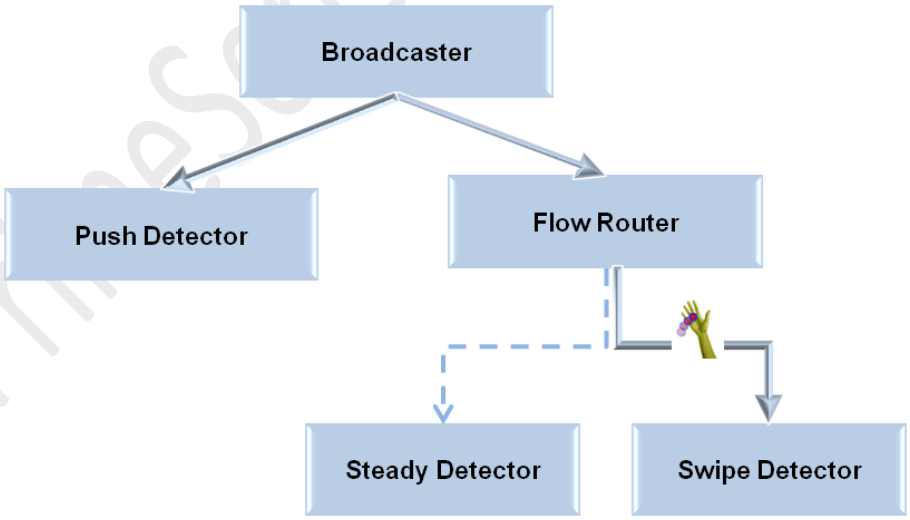

Figure 7: NITE Tree of Box Control

PrimeSense Confidential

2.5 Creating New Controls

New controls can be created based on existing messages, or on the events that the basic controls support.
For example, the “PointViewer” sample receives hand point messages. In each frame, this control saves a fixed number of points in a history buffer for each point, and draws them.

2.6 Multi-Process Support

Multi-process support is provided by two objects, encapsulating a single server and multiple clients:

1. XnVMultiProcessFlowServer object is a NITE control that receives messages and updates a shared memory section.
2. Each client is an XnVMultiProcessFlowClient object that can be used as the head of the NITE tree instead of the Session Manager. It reads from the shared memory section and sends events to all its registered listeners.

Multi-process support supplies the client with points and session events only. In addition, all communication is one-way: Server to client only. Client cannot end the session in the server without additional implementation by the user.

2.7 Troubleshooting

NITE uses the OpenNI Log mechanism, and defines the following new masks:

- XNV_NITE_MASK_CREATE: Logs the creation and destruction of Generators and Listeners.
- XNV_NITE_MASK_FLOW: Logs any message sent by any Generator and any message received by any Listener.
- XNV_NITE_MASK_CONNECT: Logs the connection and disconnection of Listeners to/from Generators and the change of active controls in Flow Routers.
- XNV_NITE_MASK_POINTS: Logs the creation and destruction of points.
- XNV_NITE_MASK_SESSION: Logs the changes of the Session state.
- XNV_NITE_MASK_MT_QUEUE: Logs access to the multi-thread queue in multi-threaded mode.
- XNV_NITE_MASK_EVENTS: Logs control events.

PrimeSense Confidential

3 NITE Samples

This chapter describes the samples that are provided with NITE. The source code of these samples is provided in the NITE package.

3.1 SingleControl Sample

The example below demonstrates the identification of wave gestures based on hand points. The output is all textual. Perform the 'click' gesture to activate, and then perform waves and see them detected.

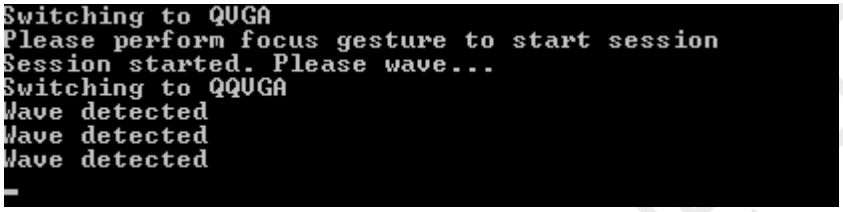

Figure 8: SingleControl output sample

3.2 CircleControl Sample

The example below depicts the identification of circular motions. A circle is drawn as the output of a circular motion. Perform the 'wave' gesture to activate, and then perform a circle to see the circle's graphic representation. A circle will appear after a full circle is detected.

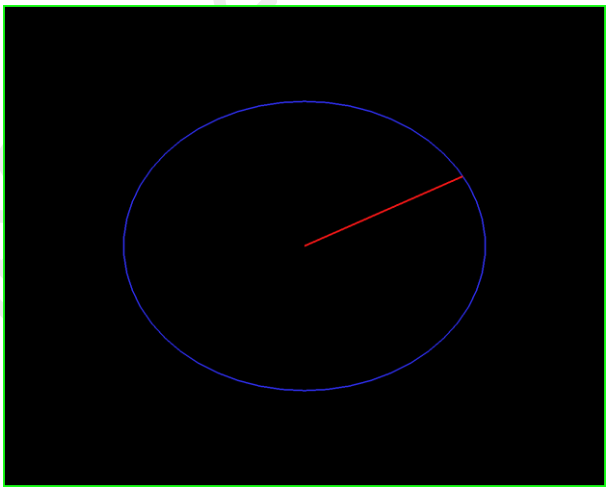

Figure 9: Circle Control Sample

3.3 PointViewer Sample

The example below demonstrates the tracking of a hand point. Perform either the 'wave' gesture or the 'click' gesture to activate, and then the hand will be tracked

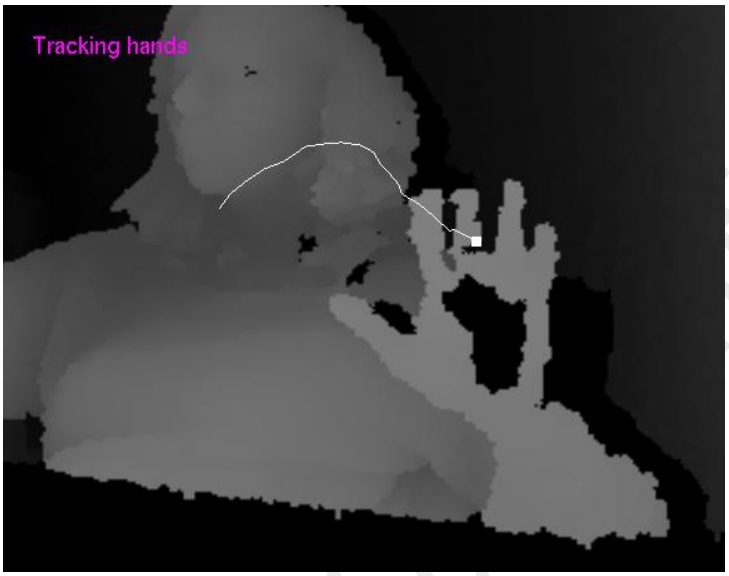

Figure 10: PointViewer Sample

3.4 Boxes Sample

The example below depicts the definition of user-specific listeners and a user-defined flow. Perform the 'wave' gesture to activate. The box control is a control that searches for a push hand gesture, and in parallel, for either a swipe or a steady hand gesture.

- A swipe gesture highlights one of the Box's walls, according to the swipe direction. A swipe up highlights that upper wall; a swipe left colors the left wall etc.
- A steady detection is used to detect any resting between swipe movements, in order to start listening to swipe again.
- Detection of a push gesture at any time results in exiting the box control.

PrimeSense Confidential

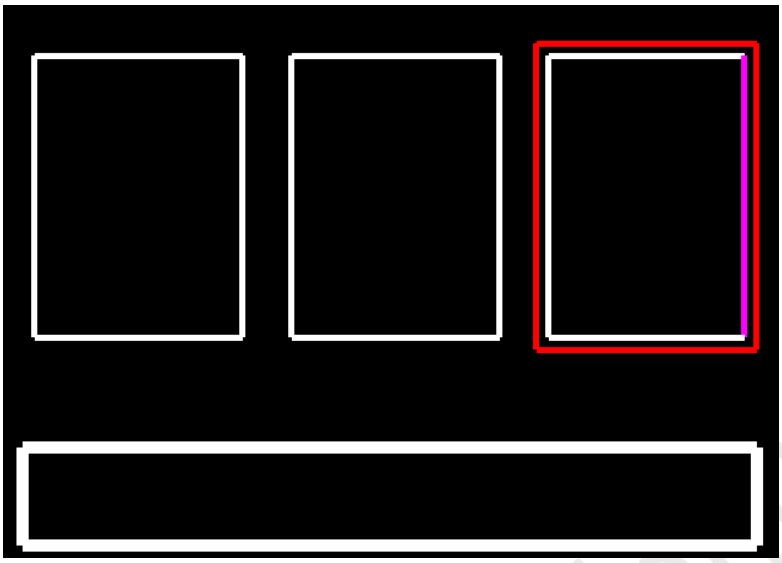

Figure 11: Boxes Sample

PrimeSense Confidential