NonVerbal Behavior Generator

Overview

The NonVerbal-Behavior-Generator (NVBG) is the module that generates behavior other than speech, such as gestures, facial expressions and gazes. The NVBG generates these behaviors based on the speech input and other messages that it receives. Nonverbal behaviors augment and emphasize spoken communication. Based on who speaks and who listens, the NVBG can characterize the NPC as a listener, speaker or a bystander and generate appropriate behavior. These behaviors are configurable using xml files provided as input to NVBG. Using these xml files, we can specify which words or parts-of-speech trigger which animations. We can specify idle animations and idle gazes that get triggered when the character is idle for a specified amount of time.

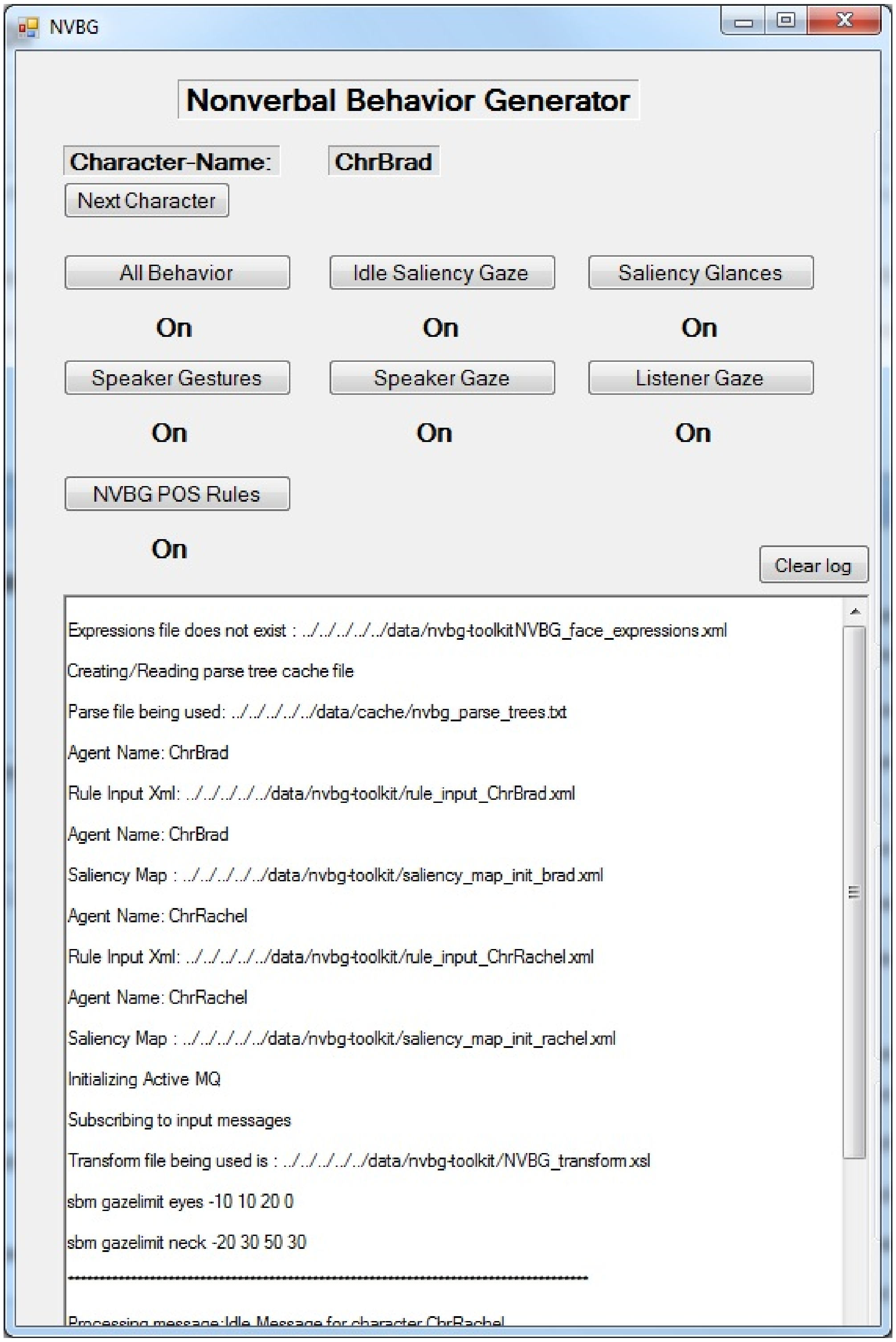
The NVBG is loosely coupled with a "parser" which gets the input sentence from NVBG and returns the parts of speech in the sentence. NVBG can extract information from the lexical, syntactic, and semantic structure of the surface text that can support the generation of believable nonverbal behaviors.

The NVBG also has a saliency map which it uses to keep track of it's environment, important events and conversational topics. This helps the NVBG generate behaviors that are appropriate to the current scene and helps it prioritize. This saliency map is configurable through xml files too.

The NVBG also allows us to configure facial expressions for a particular character using FACS units and then based on xml tags in the input messages, can trigger those facial animations.

Quick facts:

- Location: \core\NVBG
- Language: C#
- Distribution: binary-only
- Platform(s): Windows
- Main paper: Jina Lee and Stacy Marsella, "Nonverbal Behavior Generator for Embodied Conversational Agents", in 6th International Conference on Intelligent Virtual Agents, 2006, pp. 243-255. link



```
Expressions file does not exist : ../../../../../data/nvbg-toolkit NVBG_face_expressions.xml

Creating/Reading parse tree cache file

Parse file being used: ../../../../../data/cache/nvbg_parse_trees.txt

Agent Name: ChrBrad

Rule Input Xml: ../../../../../data/nvbg-toolkit/rule_input_ChrBrad.xml

Agent Name: ChrBrad

Saliency Map : ../../../../../data/nvbg-toolkit/saliency_map_init_brad.xml

Agent Name: ChrRachel

Rule Input Xml: ../../../../../data/nvbg-toolkit/rule_input_ChrRachel.xml

Agent Name: ChrRachel

Saliency Map : ../../../../../data/nvbg-toolkit/saliency_map_init_rachel.xml

Initializing Active MQ

Subscribing to input messages

Transform file being used is : ../../../../../data/nvbg-toolkit/NVBG_transform.xsl

sbm gazelimit eyes -10 10 20 0

sbm gazelimit neck -20 30 50 30

*******************************************************************************

Processing message:Idle Message for character ChrRachel
```

Users

The NVBG now supports multiple characters within one process. You can toggle between these characters as desired.The character itself needs to be configured using the various configuration options that NVBG provides. You can do this using a config file or through VHMessages.

In order to configure NVBG for a particular character, you need to have/setup the following:

- The rule_input_[culture].xml (behavior) file, that specifies which rules should be generated for parts of speech in the spoken sentence and also which animations map to which words if any.
- The .xslt transform files which process the the intermediate xml generated by the module and generate the final bml to be output. The defaults are available at http://svn.ict.usc.edu/svn_vh/trunk/data/nvbg-common/. The relevant files are NVBG_transform.xsl, NVBG_rules.xsl and NVBG_behavior_description.xsl

- The command line parameters to NVBG
- OPTIONAL - we can also specify a saliency map and a facial-expressions configuration if needed.
    1. The saliency map specifies which objects in the scene are of particular importance which allows NVBG to generate idle gazes appropriately.
    2. The facial-expressions configuration file allows us to specify which facs units should be triggered with a certain weight so that a facial expression is achieved. This facial expression can later be tagged in the input text as markup to trigger the facial expression.

Using Command Line Parameters

Required:

- **create_character [char name] [config-filename] -** You can specify multiple characters one after the other
- **data_folder_path:** Specifies the folder path to the xslt files and the rule_input_[culture].xml file.

Optional:

- **write_to_file:** Specifies whether the output bml is to be sent out as a VH message or if it should be written to a file with the name given here. Default is "false."
- **write_to_file_path:** Path of the file to be written with **write_to_file**.
- **parsetree_cachefile_path:** Specifies the file path to the file used for caching the response from the parser used by NVBG. If this doesn't exist, it is created.
- **hide_GUI:** Hides the GUI if set as true.
- **storypoint:** Specifies which story-point should be loaded from the saliency-map xml and the saliency map is updated accordingly with priorities to appropriate pawns etc.
- **-writelog:** (no arguments) If specified, NVBG will write all debugging messages to log.txt in the same folder as the binary.

Configuring the Character

You can use a config file which you can specify as a command line argument to NVBG, in order to define a character. The structure of the config file is as below:

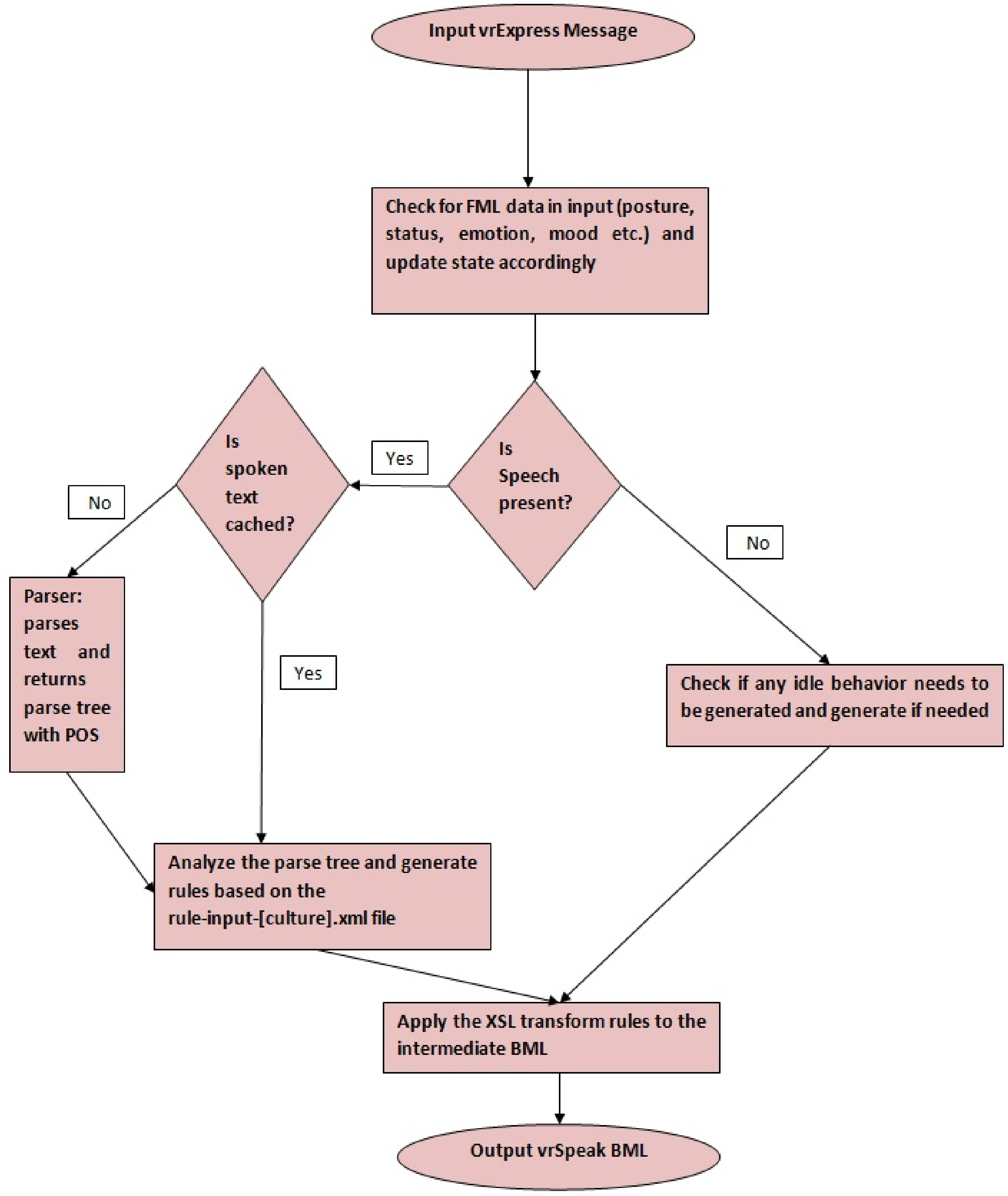
```
****************ChrBrad.ini********************

[general]
rule_input_file=rule_input_ChrBrad.xml
posture=ChrBrad@Idle01
all_behavior=on
saliency_glance=on
saliency_idle_gaze=on
speaker_gaze=on
listener_gaze=on
nvbg_POS_rules=on
saliency_map=saliency_map_init_brad.xml
```

As we can see all the information required to configure a character can be specified in the config file.

Developers

Functional Block Diagram


Input vrExpress Message
Check for FML data in input (posture, status, emotion, mood etc.) and update state accordingly
Is Speech present?
Yes
No
Is spoken text cached?
No
Yes
Parser: parses text and returns parse tree with POS
Check if any idle behavior needs to be generated and generate if needed
Analyze the parse tree and generate rules based on the rule-input-[culture].xml file
Apply the XSL transform rules to the intermediate BML
Output vrSpeak BML

Configuring Rules

The rule-input-[culture].xml file contains the rules that map certain words or parts of speech to animations. You can also specify any idle animations that you want to trigger when the character is idle for a certain period of time.

The xml code below shows how we can map a word to multiple animations which are then picked by random by NVBG when that word is spoken.

```
<rule keyword="statement_animation" priority="2" >
<pattern>is</pattern>
<pattern>are</pattern>
<pattern>were</pattern>
<pattern>was</pattern>
<pattern>have been</pattern>
<pattern>has been</pattern>
<pattern>at</pattern>
<pattern>stands</pattern>
<pattern>come</pattern>
<pattern>like</pattern>
<animation>
<posture name="CrossedArms">
<clip>CrossedArms_RArm_GestureYou02</clip>
<clip>CrossedArms_RArm_GestureYouPalmUp</clip>
</posture>
<posture name="HandsAtSide">
<clip>HandsAtSide_RArm_GestureOffer</clip>
<clip>HandsAtSide_RArm_LowBeat</clip>
<clip>HandsAtSide_RArm_MidBeat</clip>
<clip>HandsAtSide_Arms_Beat</clip>
<clip>HandsAtSide_Arms_Chop</clip>
<clip>HandsAtSide_RArm_Chop</clip>
<clip>HandsAtSide_RArm_FistsChop</clip>
<clip>HandsAtSide_RArm_LowBeat</clip>
<clip>HandsAtSide_RArm_MidBeat</clip>
</posture>
<posture name="HandsOnHip">
<clip>HandsOnHip_RArm_MidBeat</clip>
</posture>
<posture name="LHandOnHip">
<clip>LHandOnHip_RArm_You</clip>
<clip>LHandOnHip_RArm_GestureOffer</clip>
</posture>
<posture name="Chair">
<clip>Chair_You_Small</clip>
<clip>Chair_You</clip>
</posture>
</animation>
</rule>
```

The pattern <tag> contains the word that is to be matched, and the <clip> tags contain the animations which should be played when that word is spoken. The <posture> tag is used to set animations for each character posture. Only animations that are in the current characters posture will be played by NVBG. The 'priority' attribute allows NVBG to prioritize between animations when multiple one's overlap.

The rules for parts of speech are similar as shown below.

```
<rule keyword="first_NP" priority="5" >
<pattern>first_NP</pattern>
</rule>
```

```
<rule keyword="noun_phrase" priority="5" >
<pattern>NP</pattern>
</rule>
```

Notice that, in the above case, no animations are specified (although they could be). In this case, the NVBG checks the spoken sentence for parts of speech (first_NP, noun_phrase etc.) and inserts place-holder xml tags with the keyword as specified in the rule. Later when the NVBG applies the XSL transform to the intermediate BML(with the placeholder tags), it generates output behavior based on what the XSL rules specify (More on this later under the "**POS Transform rules**" section).

So basically the rule input file is a collection of these rules that the NVBG parses at runtime. Based on whether animations are directly specified in this file as <patterns> or whether they are specified further down the pipeline in the XSL rules, the NVBG generates appropriate behavior.

If you want idle animations to be generated, you can specify idle_animation rules as below

<!-- idle animation behavior -->

```
<rule keyword="idle_animation" priority="1" >
<pattern>idle_animation</pattern>
<animation>
<posture name="ChrUtah@IdleHandsAtSide01">
<clip>ChrUtah@IdleHandsAtSide01_FidgetWeightShiftRt01</clip>
<clip>ChrUtah@IdleHandsAtSide01_FidgetWeightShiftLf01</clip>
<clip>ChrUtah@IdleHandsAtSide01_FidgetHead01</clip>
</posture>
</animation>
</rule>
```

The above idle_animation rule is parsed just like the other rules and is inserted in the output bml when the character is idle.

Configuring the Saliency Map

The saliency map specifies which objects/characters/pawns in the environment are important to the character and in what priority. These priorities can vary based on what story-point we are at i.e. certain objects become important only later in a scene.

Based on the priorities of these objects, the saliency map generates idle gazes and other actions based on spoken sentences and events.

Below is an example of a saliency map for use with a character

<?xml version="1.0" encoding="utf-8" ?>

```
<storyPoints>
 <storyPoint name = "toolkitsession">
  <SaliencyMapInit>
   <pawn name = "user" recency = "0" primacy = "3"/>
   <pawn name = "upright" recency = "0" primacy = "2"/>
   <pawn name = "upleft" recency = "0" primacy = "1"/>
  </SaliencyMapInit>
  <keywordToObjectMap>
   <keyword name = "I">
    <pawn name = "upleft" primacy = "10"/>
    <pawn name = "upright" primacy = "10"/>
    <pawn name = "user" primacy = "3"/>
   </keyword>
  </keywordToObjectMap>
  <emotionInit name = "normal"/>
 </storyPoint>
</storyPoints>
```

As we can see, the saliency map can contain multiple story-points each specifying which objects in the scene are of importance for that point in the story or scene. So if NVBG is notified of a change in the scene/story, it can update the saliency map accordingly. The above example contains only one story-point but in general it can contain many. NVBG can be notified of which story-point it should load so that the appropriate priorities are assigned to the objects in the scene.

The relation between spoken words and objects in the scene can be specified with the <keywordToObjectMap> block, which contains the keyword and specifies which pawns map to it. When the character speaks those words, NVBG will make him/her gaze towards the pawn. This generates the impression that the character is aware of his environment.

You can also specify the emotion of the character using the <emotionInit> tag as shown. This emotion value can be used to change behaviors of the character, such as idle animations, facial expressions etc.

You can specify the saliency map when configuring the character. Please refer to the section above for configuring a character.

Message API

Sends:

- vrSpeak

nvbg_set_option

These are control messages to set options for NVBG. They are as shown below:

> ***nvbg_set_option [char-name] all_behavior true/false*** *- sets/unsets flag that allows all behavior generated by NVBG.*
>
> ***nvbg_set_option [char-name] saliency_glance true/false*** *- sets/unsets flag that allows saliency map generated gazes. These gazes are based on content in the speech tag and the information in the saliency map.*
>
> ***nvbg_set_option [char-name] saliency_idle_gaze true/false*** *- sets/unsets flag that allows idle gazes generated by the saliency map. These idle gazes are based on the priority of pawns in the saliency map and are generated when the character is idle.*
>
> ***nvbg_set_option [char-name] speaker_gaze true/false*** *- sets/unsets flag that allows for the character to look at the person he's speaking to.*
>
> ***nvbg_set_option [char-name] speaker_gesture true/false*** *- sets/unsets flag that allows gestures to be generated when speaking.*
>
> ***nvbg_set_option [char-name] listener_gaze true/false*** *- sets/unsets flag that allows the listener to gaze at the speaker when he speaks.*
>
> ***nvbg_set_option [char-name] nvbg_POS_rules true/false*** *- sets/unsets flag that allows behavior to be generated based on parts of speech returned by the parser.*
>
> ***nvbg_set_option [char-name] saliency_map [filename]*** *- lets you dynamically specify the saliency map that the character will use*

> ***nvbg_set_option [char-name] rule_input_file [filename]*** *- lets you dynamically specify the behavior map that the character will use*

Receives:

- vrExpress

Known Issues

FAQ

See the Main FAQ. Please use the Google Groups emailing list for unlisted questions.