

Appendix A

Taxonomy of a Virtual Identity

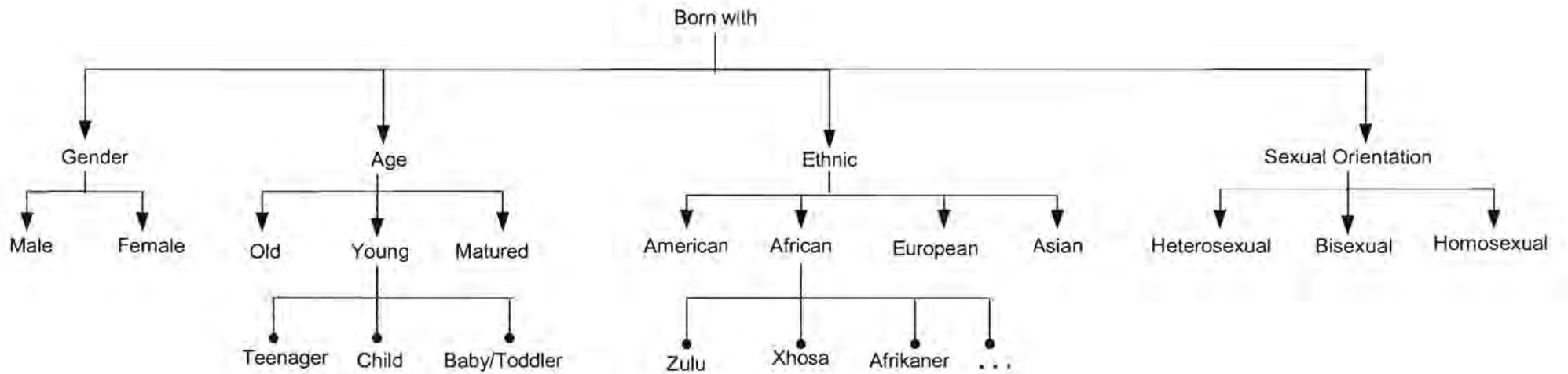
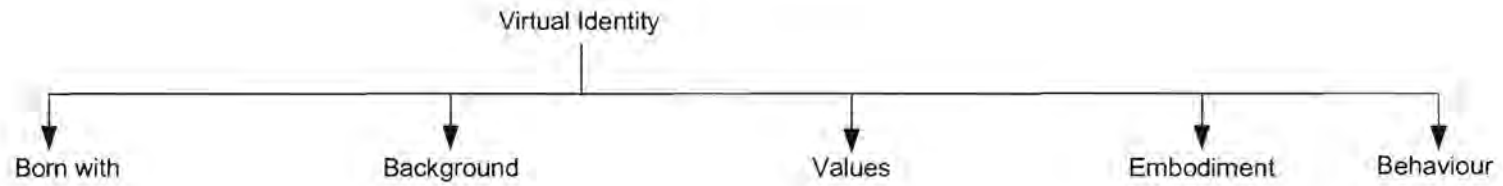
This section gives an illustration of the taxonomy that is used to define a virtual identity (refer to section 3.4).

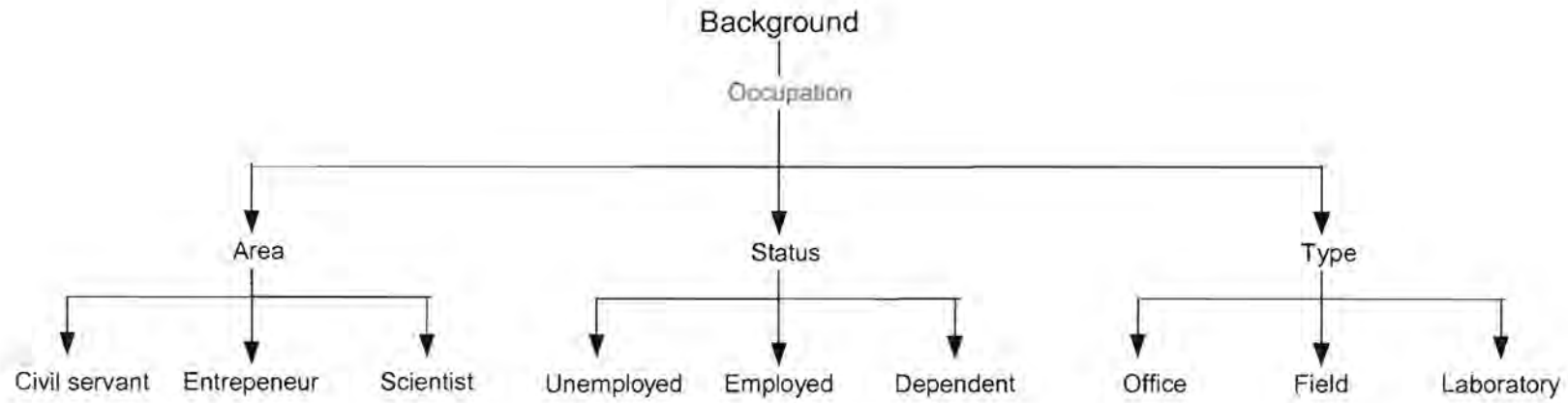
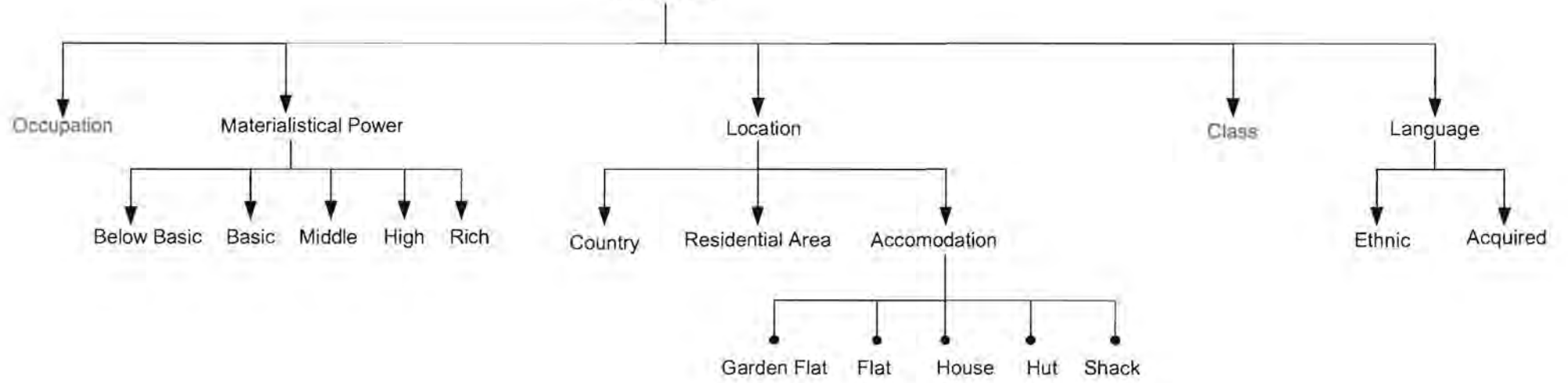
A virtual identity is defined by the following main features

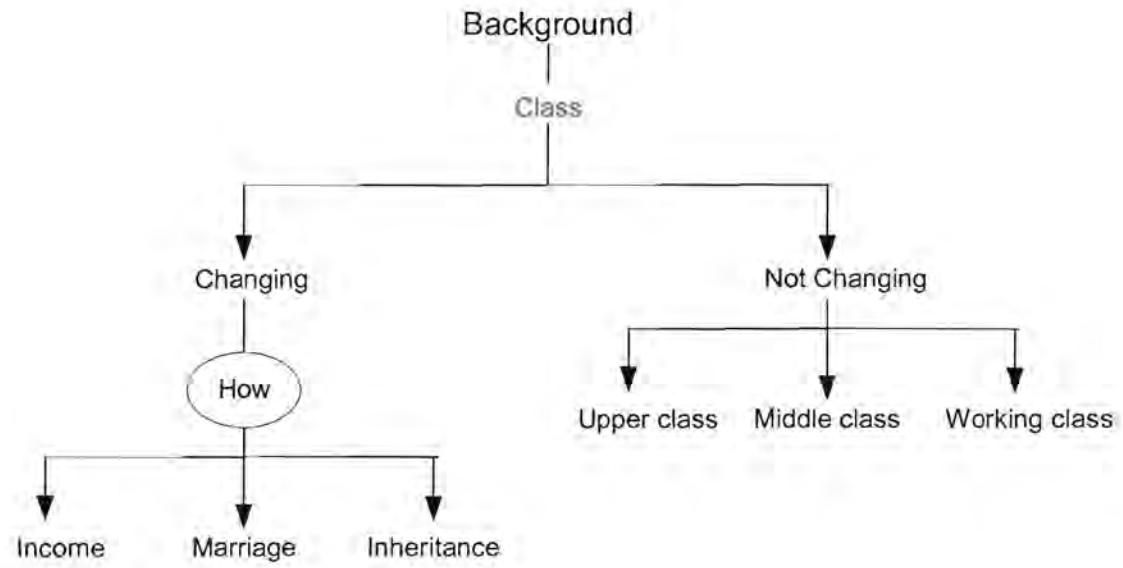
- Characteristics that a virtual identity is born with
- Characteristics concerning the virtual identity's background
- Embodiment of the virtual identity
- Behavioural characteristics of the virtual identity

Each feature is expanded into sub features.

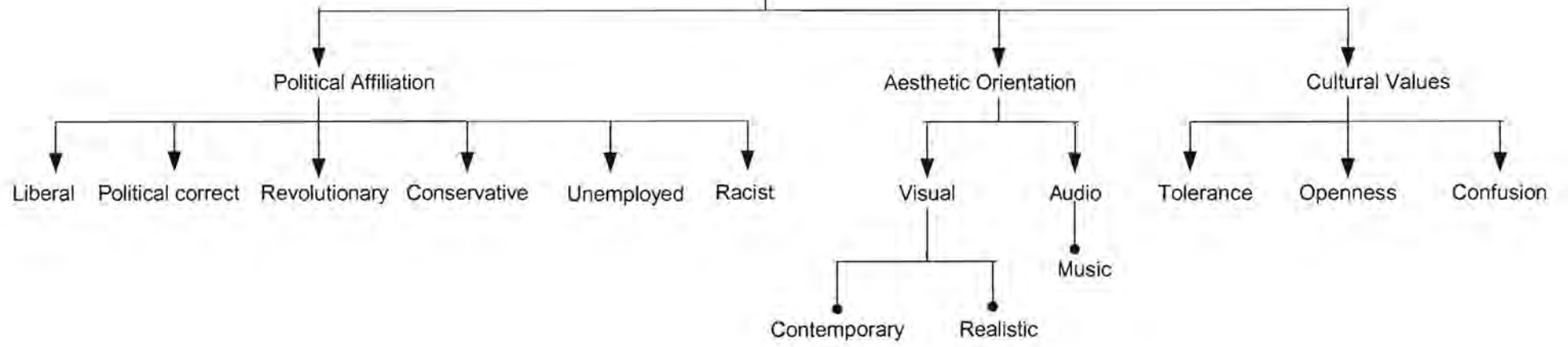
This taxonomy is by no means complete, but it can easily be extended to the level of detail that is required for a specific application.



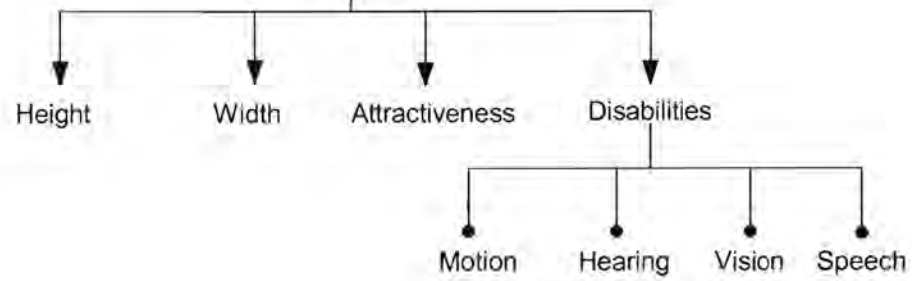


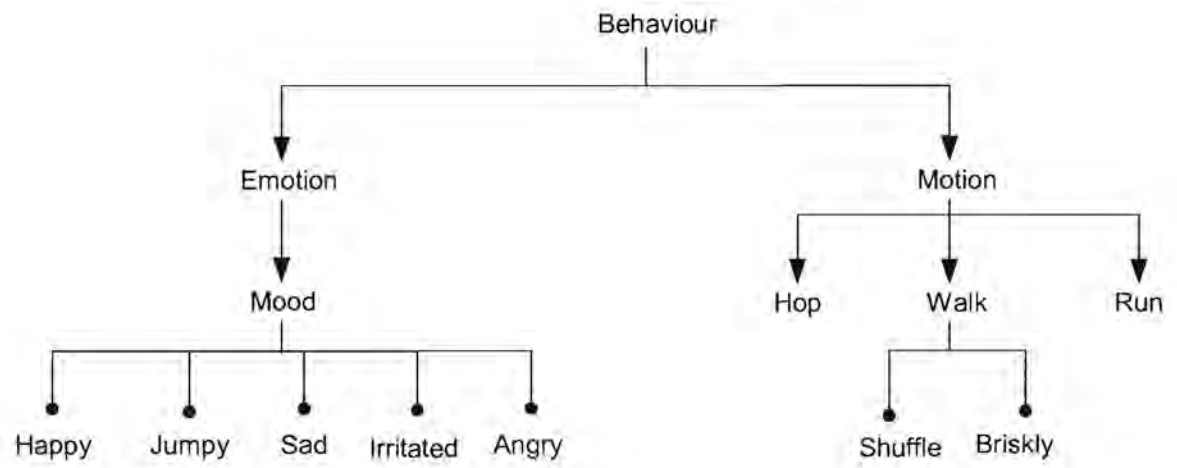


values



Embodiment







Appendix B

Interactive Storytelling Tool Code

B1. C++ node: *fpVirtualIdentity*

Section B1.1 provides the code of the header file and section B1.2 provides the code of the C++ file.

B1.1. Header file: *fpVirtualIdentity.h*

```
//fpVirtualIdentity.h defines a virtual identity

#ifndef FP_VIRTUALIDENTITY_H
#define FP_VIRTUALIDENTITY_H

#include <libfp/performer/fpDCS.h>
#include <extensions/moving/fpFlyer.h>

typedef fpSingleField<fpLink<fpFlyer>> fpSFFlyer;

class fpVirtualIdentity: public fpDCS
{
    FP_FC_DECLARE();

public:

    fpVirtualIdentity();
    ~fpVirtualIdentity();

    //time base
    fpSFDouble TimeIn;

    //fields
    fpSFFloat Age;
    fpSFBool Sex; //true = female, false = male
    fpSFString Mood;
    fpSFString Occupation;

    //input matrix for connecting nodes
    fpSFMatrix MatrixIn;
```



```
//output matrix for connecting nodes is the Matrix field

//flag to determine whether the path must be blocked or not
fpSFBool Blocked;

//flag to determine whether the person is drunk or not
fpSFBool Drunk;

//need flyer to get the value of the av-mover matrix at the current
//position of the user to do the translation
//link to the current position of the av-mover (fpFlyer)
fpSFFlyer Flyer;

//user info for connecting to the viewer
fpSFFloat UserHeight;
fpSFFloat UserBelly;
fpSFFloat Speed;
fpSFFloat Sensitivity;

//fields for the swaying effect
//time base
fpSFFloat TimeIn;
//frequency and amplitude
fpSFFloat Frequency;
fpSFFloat Amplitude;

//methods
void evaluate();
void fieldHasChanged(fpField& field);

//performer needs this
static pfType* getClassType() {return classType;}

//try to find a path in pfMemory in the inheritance graph /*virtual*/
pfMemory* castToPerformer();

protected:
```

private:

```
//this is for Performer rtti
static pFType* classType;

//to determine whether the Age or Drunk field is activated or not
int _age;
int _drunk;

//related to the age field - sets the viewpoint according to the age
float _height;

fpLink<fpFlyer> _my_flyer;
};
#endif
```

B1.2. C++ file: *fpVirtualIdentity.c++*

```
// *****
// * This node defines the virtual identity in general      *
// * It defines the fields required by all virtual identities *
// * It only changes the viewpoint                          *
// * Other virtual identities inherit from this class       *
// * and then specializes in the behaviour                 *
// *                                                         *
// *****

#include "fpVirtualIdentity2.h"
#include <extensions/moving/fpFlyer.h>

FP_FC_DEFINE(fpVirtualIdentity)

pFType* fpVirtualIdentity::classType = NULL;
```

```

//constructor
fpVirtualIdentity::fpVirtualIdentity():
    _height(0.0)
{
    //Performer type init
    setType(classType);

    FP_FC_START_CONSTRUCT(fpDCS);

    fpMatrix identity;
    identity.makeIdent();

    //initializing fields

    FP_FC_ADD_FIELD(Flyer,    fpLink<fpFlyer>());
    FP_FC_ADD_FIELD(Age,     0.0);
    FP_FC_ADD_FIELD(Sex,     TRUE);
    FP_FC_ADD_FIELD(Mood,    "happy");
    FP_FC_ADD_FIELD(Occupation, "unknown");
    FP_FC_ADD_FIELD(Speed,   0.0);
    FP_FC_ADD_FIELD(Sensitivity, 0.0);
    FP_FC_ADD_FIELD(Blocked,  FALSE);
    FP_FC_ADD_FIELD(Drunk,   FALSE);
    FP_FC_ADD_FIELD(MatrixIn, identity);
    FP_FC_ADD_FIELD(UserHeight, 0.0);
    FP_FC_ADD_FIELD(UserBelly, 0.0);
    FP_FC_ADD_FIELD(TimeIn,   0.0);
    FP_FC_ADD_FIELD(Frequency, 5.0);
    FP_FC_ADD_FIELD(Amplitude, 0.3);

    FP_FC_FINISH_CONSTRUCT();
}

//destructor
fpVirtualIdentity::~fpVirtualIdentity()
{
}

```

//methods that are always required by Performer

```

void fpVirtualIdentity::initClass()
{
if (classType == NULL)
{
fpSFFlyer::initClass("fpSFFlyer", "fpField");
classType = new pfType(fpDCS::getClassType(), "fpVirtualIdentity");

FP_FC_INIT(fpDCS, fpVirtualIdentity, TRUE, classType);
}
}

/* virtual */
pfMemory* fpVirtualIdentity::castToPerformer()
{
return (pfMemory*)this;
}

/* virtual */
void fpVirtualIdentity::evaluate()
{
//if view has not been changed according to age
if (_age)
{
/*_my_flyer = Flyer.getValue();

fpMatrix tmp_mat;
tmp_mat.copy(_my_flyer->Matrix.getValue()); */

if (Age.getValue() <= 12)
{
_height = 1.0;
UserHeight.setValue(1.0);
UserBelly.setValue(0.5);
Speed.setValue(1.0);
Sensitivity.setValue(0.02);
}
}
}

```

```
Blocked.setValue(TRUE);
}
else if (Age.getValue() > 12)
{
    _height = 1.8;
    UserHeight.setValue(1.8);
    UserBelly.setValue(0.9);
    Blocked.setValue(FALSE);

    if (Age.getValue() < 60)
    {
        Speed.setValue(1.0);
        Sensitivity.setValue(0.02);
    }
    else
    {
        Speed.setValue(0.0001);
        Sensitivity.setValue(0.001);
    }
}

_age=0;

}
else
{
    _my_flyer = Flyer.getValue();

    fpMatrix tmp_mat;
    tmp_mat.copy(_my_flyer->Matrix.getValue());

    fpMatrix new_mat;
    cerr << "In the else loop ... \n" << flush;

    new_mat.makeIdent();
    new_mat.copy(tmp_mat);
    new_mat.makeTrans(new_mat[3][0], new_mat[3][1], _height);
```

```
if(!_drunk)
{
    //to calculate the translation with the swaying
    fpVec3 _position;

    _position[0] = sin(Amplitude.getValue()) * sin(TimeIn.getValue() *
        Frequency.getValue());
    new_mat.makeTrans(_position[0], new_mat[3][1], new_mat[3][2]);
}

Matrix.setValue(new_mat);
}
}

/* virtual */
void fpVirtualIdentity::fieldHasChanged(fpField &field)
{
    if(&field == &Age)
        _age = 1;

    if(&field == &Drunk)
        _drunk = 1;
}
```



B2. Scheme Scripts

Section B2.1 presents the scheme script for the shebeen owner. Section B2.2 presents the scheme script for the Zulu man and section B2.3 presents the scheme script for the Zulu boy. Section B2.4 presents the scheme script with the different objects that have hooks and that are loaded into the first three scheme scripts.

B2.1. Shebeen owner

```
;; *****
;;
;; *
;; * A sound file is played when the user clicks on the drum
;; * To run the program use the following:
;; * 1. Start the sound service with sndserv.sh -p <filepath>
;; * 2. Run the script with the following command:
;; * /home/medicine/avocado/bin/aview.sh -o sndserv:quitte
;; *
;; *****
```

```
(define (mono) (for-each (lambda (scr) (-> (-> scr 'StereoOn) 'set-value 0)) av-all-screens))
(define (stereo) (for-each (lambda (scr) (-> (-> scr 'StereoOn) 'set-value 1)) av-all-screens))
```

```
;; drag tool to drag around objects, which hold compatible draggers
```

```
(define drag-tool-right (make-instance-by-name "fpDragTool"))
; the point tool
(-> (-> drag-tool-right 'Name) 'set-value "drag-right")
(-> (-> drag-tool-right 'TimeIn) 'connect-from (-> time-sensor 'Time))
```

```
(define drag-tool-right-geo (make-instance-by-name "fpLoadFile"))
(-> (-> drag-tool-right-geo 'Filename) 'set-value (in-vicinity av-data-vicinity "av-stylus.iv"))
(-> (-> drag-tool-right 'Cursor) 'set-value drag-tool-right-geo)
(-> (-> drag-tool-right 'ToolManager) 'set-value av-tool-manager)
```

```
;; the drag tool becomes the default tool for the stylus
(-> (-> av-stylus 'DefaultTool) 'set-value drag-tool-right)
(-> (-> av-stylus 'Tool) 'set-value drag-tool-right)
```

```
;;defining geometry
```

```
;;(av-set-tool-dragger-match "fpPickTool" "fpScriptDragger"  
"fpPickToolScriptDraggerCon")
```

```
;(fp-set-value av-stylus 'DefaultTool dragTool)
```

```
;(fp-set-value av-stylus 'Tool dragTool)
```

```
(define room(make-instance-by-name "fpLoadFile"))
```

```
(fp-set-value room 'Name "room")
```

```
(fp-set-value room 'Filename
```

```
"/home/marde/avango/GMD/virtual_identity/shebeen/shebeen.flr")
```

```
(fp-set-value room 'Matrix (mult-mat (make-rot-mat -90 0 0 1)(make-trans-mat 1 4 0)))
```

```
(av-add room)
```

```
;;define the virtual identity
```

```
(define identity(make-instance-by-name "fpVirtualIdentity"))
```

```
(fp-set-value identity 'Flyer av-mover)
```

```
(fp-set-value identity 'Age 30)
```

```
(fp-set-value identity 'Sex 0)
```

```
(fp-set-value identity 'Mood "Happy")
```

```
(av-add identity)
```

```
(fp-set-value av-mover 'CollisionResponse 1)
```

```
;;connect the virtual identity to the viewer
```

```
(fp-connect-from av-mover 'Matrix identity 'Matrix)
```

```
(fp-connect-from av-viewer 'Matrix av-mover 'Matrix)
```

```
(fp-connect-from av-mover 'UserHeight identity 'UserHeight)
```



```
(fp-connect-from av-mover 'UserBelly identity 'UserBelly)
(fp-connect-from av-mover 'Sensitivity identity 'Sensitivity)
(fp-connect-from av-mover 'Speed identity 'Speed)
(fp-set-value av-mover 'Acceleration 0.1)
(fp-set-value av-mover 'CollisionResponse 1)
```

```
::=====
```

```
::get hooks to objects
```

```
(load "./shebeenPieces.scm")
```

```
(define radioDragger(make-instance-by-name "fpScriptDragger"))
(fp-set-value radioDragger 'PushCB "(sound-cb)")
```

```
(define nliste '())
```

```
(for-each
  (lambda (piece-name)
    (let
      (
        (piece (-> room 'get-hook fp-dcs piece-name))
        (moveDragger (make-instance-by-name "fpMatrixDragger"))
      )
      (display piece-name) (newline)
      (if piece
        (begin
          (if (equal? piece-name "dcs_button_top")
            (begin
              (fp-add-1 value piece 'Dragger radioDragger)
              (fp-set-value piece 'Name "piece-name")
              (display "Dragger connected to ")
              (display piece-name) (newline)
            ))
          (if
            (or
              (equal? piece-name "dcs_table_up")
              (equal? piece-name "dcs_table_ground")
```

```

(equal? piece-name "dcs_crate_1")
(equal? piece-name "dcs_crate_2")
(equal? piece-name "dcs_crate_3")
(equal? piece-name "dcs_crate_4")
(equal? piece-name "dcs_chair_1")
(equal? piece-name "dcs_chair_2")
(equal? piece-name "dcs_chair_3")
(equal? piece-name "dcs_chair_4")
(equal? piece-name "dcs_crate_table_1")
(equal? piece-name "dcs_crate_table_2")
(equal? piece-name "dcs_crate_5")
(equal? piece-name "dcs_crate_6")
(equal? piece-name "dcs_crate_7")
(equal? piece-name "dcs_crate_8")
(equal? piece-name "dcs_cushion_1")
(equal? piece-name "dcs_cushion_2"))
(begin
  (fp-add-l value piece 'Dragger moveDragger)
  (fp-connect-from piece 'Matrix moveDragger 'Matrix)
  (fp-set-value piece 'Name "piece-name")
  (display "Dragger connected to ")
  (display piece-name) (newline)
))
)
)
)
)
room-pieces
)

;;=====
;;define sounds

(av-assert-extension 'av-ext-sound)

;;initialize the sound server
(av-sndserv-reset)

```

```
(av-sndserv-on)
```

```
;;create a sound source
```

```
(define sound-source (make-instance-by-name "fpSoundSource"))
```

```
;; set the name of the sample to play
```

```
(fp-set-value sound-source 'EarNode av-mover)
```

```
(fp-set-value sound-source 'SourceMode "f1")
```

```
(fp-set-value sound-source 'PlayMode "co")
```

```
(fp-set-value sound-source 'SpatMode "dy")
```

```
(fp-set-value sound-source 'SoundName "shibeen_radio")
```

```
(fp-connect-from sound-source 'TimeIn time-sensor 'Time)
```

```
;;
```

```
;;callback to play sound when the radio is selected
```

```
(define (sound-cb)
```

```
  ;; play the sound sample
```

```
  (display "playing...") (newline)
```

```
  (fp-set-value sound-source 'Playing 1)
```

```
)
```

B2.2. Zulu man

```
.. *****
;;
.. *
;; *
;; * A sound file is played when the user clicks on the drum
;; *
;; * To run the program use the following:
;; *
;; * 1. Start the sound service with sndserv.sh -p <filepath>
;; *
;; * 2. Run the script with the following command:
;; *
;; * /home/medicine/avocado/bin/aview.sh -o sndserv:quitte
;; *
.. *****
```

```
(define (mono) (for-each (lambda (scr) (-> (-> scr 'StereoOn) 'set-value 0)) av-all-screens))
```

```
(define (stereo) (for-each (lambda (scr) (-> (-> scr 'StereoOn) 'set-value 1)) av-all-screens))
```

```
;; drag tool to drag around objects, which hold compatible draggers
```

```

(define drag-tool-right (make-instance-by-name "fpDragTool"))
;; the point tool
(-> (-> drag-tool-right 'Name) 'set-value "drag-right")
(-> (-> drag-tool-right 'TimeIn) 'connect-from (-> time-sensor 'Time))

(define drag-tool-right-geo (make-instance-by-name "fpLoadFile"))
(-> (-> drag-tool-right-geo 'Filename) 'set-value (in-vicinity av-data-vicinity "av-stylus.iv"))
(-> (-> drag-tool-right 'Cursor) 'set-value drag-tool-right-geo)
(-> (-> drag-tool-right 'ToolManager) 'set-value av-tool-manager)

;; the drag tool becomes the default tool for the stylus
(-> (-> av-stylus 'DefaultTool) 'set-value drag-tool-right)
(-> (-> av-stylus 'Tool) 'set-value drag-tool-right)

;;-----
;;defining geometry

;;(av-set-tool-dragger-match "fpPickTool" "fpScriptDragger"
"fpPickToolScriptDraggerCon")

;(fp-set-value av-stylus 'DefaultTool dragTool)
;(fp-set-value av-stylus 'Tool dragTool)

(define room(make-instance-by-name "fpLoadFile"))
(fp-set-value room 'Name "room")
(fp-set-value room 'Filename
"/home/marde/avango/GMD/virtual_identity/shebeen/shebeen.flt")
(fp-set-value room 'Matrix (mult-mat (make-rot-mat -90 0 0 1)(make-trans-mat 1 4 0)))
(av-add room)

;;-----
;;define the virtual identity

(define identity(make-instance-by-name "fpVirtualIdentity"))
(fp-set-value identity 'Flyer av-mover)
(fp-set-value identity 'Age 30)

```

```
(fp-set-value identity 'Sex 0)
(fp-set-value identity 'Mood "Happy")
(av-add identity)

(fp-set-value av-mover 'CollisionResponse 1)
```

```
;;=====
;;connect the virtual identity to the viewer
```

```
(fp-connect-from av-mover 'Matrix identity 'Matrix)
(fp-connect-from av-viewer 'Matrix av-mover 'Matrix)
(fp-connect-from av-mover 'UserHeight identity 'UserHeight)
(fp-connect-from av-mover 'UserBelly identity 'UserBelly)
(fp-connect-from av-mover 'Sensitivity identity 'Sensitivity)
(fp-connect-from av-mover 'Speed identity 'Speed)
(fp-set-value av-mover 'Acceleration 0.1)
(fp-set-value av-mover 'CollisionResponse 1)
```

```
;;=====
;;get hooks to objects
```

```
(load "./shebeenPieces.scm")
```

```
(define radioDragger(make-instance-by-name "fpScriptDragger"))
(fp-set-value radioDragger 'PushCB "(sound-cb)")
```

```
(define nliste '())
```

```
(for-each
  (lambda (piece-name)
    (let
      (
        (piece (-> room 'get-hook fp-dcs piece-name))
        (moveDragger (make-instance-by-name "fpMatrixDragger"))
      )
      (display piece-name) (newline)
      (if piece
```

```
(begin
  (if (equal? piece-name "dcs_button_top")
    (begin
      (fp-add-1 value piece 'Dragger radioDragger)
      (fp-set-value piece 'Name "piece-name")
      (display "Dragger connected to ")
      (display piece-name) (newline)
    ))
  (if
    (or
      (equal? piece-name "dcs_table_up")
      (equal? piece-name "dcs_table_ground")
      (equal? piece-name "dcs_crate_1")
      (equal? piece-name "dcs_crate_2")
      (equal? piece-name "dcs_crate_3")
      (equal? piece-name "dcs_crate_4")
      (equal? piece-name "dcs_chair_1")
      (equal? piece-name "dcs_chair_2")
      (equal? piece-name "dcs_chair_3")
      (equal? piece-name "dcs_chair_4")
      (equal? piece-name "dcs_crate_table_1")
      (equal? piece-name "dcs_crate_table_2")
      (equal? piece-name "dcs_crate_5")
      (equal? piece-name "dcs_crate_6")
      (equal? piece-name "dcs_crate_7")
      (equal? piece-name "dcs_crate_8")
      (equal? piece-name "dcs_cushion_1")
      (equal? piece-name "dcs_cushion_2"))
    (begin
      (fp-add-1 value piece 'Dragger moveDragger)
      (fp-connect-from piece 'Matrix moveDragger 'Matrix)

      (fp-set-value piece 'Name "piece-name")
      (display "Dragger connected to ")
      (display piece-name) (newline)
    ))
  )
)
```

```

    )
  )
)
room-pieces
)

;;=====
;;define sounds

(av-assert-extension 'av-ext-sound)

;;initialize the sound server
(av-sndserv-reset)
(av-sndserv-on)

;;create a sound source
(define sound-source (make-instance-by-name "fpSoundSource"))

;; set the name of the sample to play
(fp-set-value sound-source 'EarNode av-mover)
(fp-set-value sound-source 'SourceMode "fl")
(fp-set-value sound-source 'PlayMode "co")
(fp-set-value sound-source 'SpatMode "dy")
(fp-set-value sound-source 'SoundName "shibeen_radio")
(fp-connect-from sound-source 'TimeIn time-sensor 'Time)

;;=====
;;callback to play sound when the radio is selected

(define (sound-cb)
  ;; play the sound sample
  (display "playing...") (newline)
  (fp-set-value sound-source 'Playing 1)
)

```

B2.3. Zulu boy

```

.. *****
;;
.. *
;; *
;; * A sound file is played when the user clicks on the drum
;; *
;; * To run the program use the following:
;; *
;; * 1. Start the sound service with sndserv.sh
;; *
;; * 2. Run the script with the following command:
;; *
;; * /home/medicine/avocado/bin/aview.sh -o sndserv:quitte
;; *
;; * -p <filepath>
;; *
.. *****
;;
;;=====
;;defining geometry

;(av-set-tool-dragger-match "fpPickTool" "fpScriptDragger"
"fpPickToolScriptDraggerCon")

(define room(make-instance-by-name "fpLoadFile"))
(fp-set-value room 'Name "room")
(fp-set-value room 'Filename
"/home/marde/avango/GMD/virtual_identity/shebeen/shebeen.flr")

(fp-set-value room 'Matrix (mult-mat (make-rot-mat -90 0 0 1)(make-trans-mat 1 4 0)))
(av-add room)

;;=====
;;creating the light source
(define light(make-instance-by-name "fpGlobalLight"))
(fp-set-value light 'Matrix (make-trans-mat 0 0 1))
(av-add light)

;;=====
;;define the virtual identity

```



```
(define identity(make-instance-by-name "fpVirtualIdentity"))
(fp-set-value identity 'Flyer av-mover)
(fp-set-value identity 'Age 6)
(fp-set-value identity 'Sex 0)
(fp-set-value identity 'Mood "Happy")
(av-add identity)
```

```
;;
```

```
;;connect the virtual identity to the viewer
```

```
(fp-connect-from av-mover 'Matrix identity 'Matrix)
(fp-connect-from av-viewer 'Matrix av-mover 'Matrix)
(fp-connect-from av-mover 'UserHeight identity 'UserHeight)
(fp-connect-from av-mover 'UserBelly identity 'UserBelly)
(fp-connect-from av-mover 'Sensitivity identity 'Sensitivity)
(fp-connect-from av-mover 'Speed identity 'Speed)
(fp-set-value av-mover 'Acceleration 0.1)
(fp-set-value av-mover 'CollisionResponse 1)
(fp-set-value av-mover 'Dof 10)
(fp-set-value av-mover 'Speed 1.0)
(fp-connect-from av-viewer 'Matrix av-mover 'Matrix)
```

```
;;
```

```
;; blocking path
```

```
(define _override-mtl (make-instance-by-name "fpOverrideMtl"))
(define _change-mtl (make-instance-by-name "fpMaterial"))
(-> (-> _override-mtl 'Material) 'set-value _change-mtl)
(-> (-> _override-mtl 'TransparencyMode) 'set-value 5)
(-> (-> _change-mtl 'FrontColorMode) 'set-value 0)
(-> (-> _change-mtl 'BackColorMode) 'set-value 0)
(av-add _override-mtl)
```

```
(define blocking_cube (make-instance-by-name "fpLoadFile"))
(fp-set-value blocking_cube 'Matrix (mult-mat
                                     (make-scale-mat 40 10 60)
                                     (make-trans-mat 10 50 0)))
```

```

(fp-add-1value _override-mtl 'Children blocking_cube)
(fp-set-value blocking_cube 'Filename
"/home/medicine/work/TestZone/graphics/iv/misc/PANEL.iv")

(-> (-> _change-mtl 'Ambient) 'set-value (make-vec3 1 1 1))
(-> (-> _change-mtl 'Alpha) 'set-value 0.0)

(fp-set-value _override-mtl 'Enable 1)

;;=====
;;getting the value of the blocking flag and determining the
;;blocked path

(define block-path-trigger (make-instance-by-name "fpTriggerCB"))
(fp-connect-from block-path-trigger 'Input identity 'Blocked)
(fp-set-value block-path-trigger 'Callback (lambda ignore (block-path-cb)))

(define (block-path-cb)
  (if (equal? (fp-get-value identity 'Blocked) 1)
      (begin
        (av-add _override-mtl)
        (display "Blocking path...")
      )
      )
  )
  (if (equal? (fp-get-value identity 'Blocked) 0)
      (begin
        (av-remove _override-mtl)
        (display "Unblocking path...")
      )
      )
  )
  )

;;=====
;;getting the value of the identity's Age and setting the draggers
;;accordingly

```

```
(define interaction-trigger (make-instance-by-name "fpTriggerCB"))
(fp-connect-from interaction-trigger 'Input identity 'Age)
(fp-set-value interaction-trigger 'Callback (lambda ignore (interaction-cb)))
```

```
;;
```

```
;;set the properties of the fpFlyer
```

```
(fp-set-value av-mover 'Dof 10)
```

```
;;lets the user be bounced back when he/she collides with an object
```

```
(fp-set-value av-mover 'CollisionResponse 1)
```

```
;;
```

```
;;define sounds
```

```
(av-assert-extension 'av-ext-sound)
```

```
;;initialize the sound server
```

```
(av-sndserv-reset)
```

```
(av-sndserv-on)
```

```
;;create a sound source
```

```
(define sound-source (make-instance-by-name "fpSoundSource"))
```

```
;; set the name of the sample to play
```

```
(fp-set-value sound-source 'EarNode av-mover)
```

```
(fp-set-value sound-source 'SourceMode "fl")
```

```
(fp-set-value sound-source 'PlayMode "co")
```

```
(fp-set-value sound-source 'SpatMode "dy")
```

```
(fp-set-value sound-source 'SoundName "shibeen_radio")
```

```
(fp-connect-from sound-source 'TimeIn time-sensor 'Time)
```

```
;;
```

```
;;get hooks to objects
```

```
(load "./shibeenPieces.scm")
```

```
(define photoDragger(make-instance-by-name "fpScriptDragger"))
```

```

(fp-set-value photoDragger 'PushCB "(photo-cb)")

(define moveDragger(make-instance-by-name "fpMatrixDragger"))

(define sitDragger(make-instance-by-name "fpScriptDragger"))
(fp-set-value sitDragger 'PushCB "(sit-cb)")

(define drumDragger(make-instance-by-name "fpScriptDragger"))
(fp-set-value drumDragger 'PushCB "(sound-cb)")

(define transitionDragger(make-instance-by-name "fpScriptDragger"))
(fp-set-value transitionDragger 'PushCB "(trans-cb)")

(fp-add-1 value blocking_cube 'Dragger transitionDragger)

(define nliste '())

;;callback to allow the correct interactions according to the identity's Age
(define (interaction-cb)
  (if (< (fp-get-value identity 'Age) 13)
      (begin
        (av-add _override-mtl)
        (display "Blocking path...") (newline)

        (for-each
         (lambda (piece-name)
           (let (
             (piece (-> room 'get-hook fp-dcs piece-name))
           )
             (if (equal? piece-name "dcs_soccer")
                 (begin
                   (fp-remove-1 value piece 'Dragger photoDragger)
                   (fp-set-value piece 'Name "piece-name")
                   (fp-set-value sound-source 'SoundName "soccer")
                   ;(fp-set-value sound-source 'SoundName "shibeen_radio")

                   (display "Dragger removed from ")
                   (display piece-name) (newline)

```

```

))
(if (equal? piece-name "dcs_boxing_photo")
  (begin
    (fp-remove-1 value piece 'Dragger photoDragger)
    (fp-set-value piece 'Name "piece-name")
    (fp-set-value sound-source 'SoundName "crowdy")
    ;(fp-set-value sound-source 'SoundName "shibeen_radio")
    (display "Dragger removed from ")
    (display piece-name) (newline)
  ))
(if (equal? piece-name "dcs_jazz")
  (begin
    (fp-remove-1 value piece 'Dragger photoDragger)
    (fp-set-value piece 'Name "piece-name")
    (fp-set-value sound-source 'SoundName "jazz_photo")
    ;(fp-set-value sound-source 'SoundName "shibeen_radio")
    (display "Dragger removed from ")
    (display piece-name) (newline)
  ))

(if
  (or
    (equal? piece-name "dcs_cup_1")
    (equal? piece-name "dcs_cup_2")
    (equal? piece-name "dcs_cup_3")
    (equal? piece-name "dcs_cup_4")
    (equal? piece-name "dcs_cup_5")
    (equal? piece-name "dcs_cup_6")
    (equal? piece-name "dcs_cup_7")
    (equal? piece-name "dcs_cup_8")
    (equal? piece-name "dcs_cup_9"))
  (begin
    (fp-remove-1 value piece 'Dragger moveDragger)
    (fp-set-value piece 'Name "piece-name")
    (display "Dragger removed from ")
    (display piece-name) (newline)
  ))

```

```

(if (equal? piece-name "dcs_crate_1")
  (begin
    (fp-remove-1 value piece 'Dragger sitDragger)
    (fp-set-value piece 'Name "piece-name")
    (display "Dragger removed from ")
    (display piece-name) (newline)
  ))
(if (equal? piece-name "dcs_crate_2")
  (begin
    (fp-remove-1 value piece 'Dragger moveDragger)
    (fp-set-value piece 'Name "piece-name")
    (display "Dragger removed from ")
    (display piece-name) (newline)
  ))
(if (equal? piece-name "dcs_crate_3")
  (begin
    (fp-remove-1 value piece 'Dragger moveDragger)
    (fp-set-value piece 'Name "piece-name")
    (display "Dragger removed from ")
    (display piece-name) (newline)
  ))
(if (equal? piece-name "dcs_crate_4")
  (begin
    (fp-remove-1 value piece 'Dragger moveDragger)
    (fp-set-value piece 'Name "piece-name")
    (display "Dragger removed from ")
    (display piece-name) (newline)
  ))
(if piece
  (begin
    (if (equal? piece-name "dcs_button_top")
      (begin
        (fp-add-1 value piece 'Dragger drumDragger)
        (fp-set-value piece 'Name "piece-name")
        (display "Dragger connected to ")

```

```

      (display piece-name) (newline)
    ))
  (if (equal? piece-name "dcs_front_wall")
      (begin
        (fp-add-lvalue piece 'Dragger transitionDragger)
        (fp-set-value piece 'Name "piece-name")
        (display "Dragger connected to ")
        (display piece-name) (newline)
      ))
    ))
  ))
  room-pieces)
)
)
(if (> (fp-get-value identity 'Age) 11)
    (begin
      (av-remove _override-mtl)
      (display "Unblocking path...") (newline)
      (for-each
        (lambda (piece-name)
          (let (
            (piece (-> room 'get-hook fp-dcs piece-name))
          )
            (if piece
              (begin
                (if (equal? piece-name "dcs_button_top")
                    (begin
                      (fp-remove-lvalue piece 'Dragger drumDragger)
                      (fp-set-value piece 'Name "piece-name")
                      (display "Dragger removed from ")
                      (display piece-name) (newline)
                    ))
                (if (equal? piece-name "dcs_front_wall")
                    (begin
                      (fp-remove-lvalue piece 'Dragger transitionDragger)
                      (fp-set-value piece 'Name "piece-name")
                    )
                )
              )
            )
          )
        )
      )
    )
  )
)

```

```
(display "Dragger removed from ")
(display piece-name) (newline)
))
(if
(equal? piece-name "dcs_soccer")
(begin
(fp-add-1value piece 'Dragger photoDragger)
(fp-set-value piece 'Name "piece-name")
(fp-set-value sound-source 'SoundName "soccer")
;(fp-set-value sound-source 'SoundName "shibeen_radio")
(display "Dragger connected to ")
(display piece-name) (newline)
))
(if (equal? piece-name "dcs_boxing_photo")
(begin
(fp-add-1value piece 'Dragger photoDragger)
(fp-set-value piece 'Name "piece-name")
(fp-set-value sound-source 'SoundName "crowdy")
;(fp-set-value sound-source 'SoundName "shibeen_radio")
(display "Dragger connected to ")
(display piece-name) (newline)
))
(if (equal? piece-name "dcs_jazz")
(begin
(fp-add-1value piece 'Dragger photoDragger)
(fp-set-value piece 'Name "piece-name")
(fp-set-value sound-source 'SoundName "jazz_photo")
;(fp-set-value sound-source 'SoundName "shibeen_radio")
(display "Dragger connected to ")
(display piece-name) (newline)
))
(if
(or
(equal? piece-name "dcs_cup_1")
(equal? piece-name "dcs_cup_2")
(equal? piece-name "dcs_cup_3")
(equal? piece-name "dcs_cup_4"))
```


)

;;callback to make transition from boy to man

(define (trans-cb)

;; change the age of the identity

(fp-set-value identity 'Age 20)

(fp-set-value av-mover 'Speed 0.5)

(fp-set-value av-mover 'Dof 1)

)

;;=====

B2.4. Shebeen pieces

;; the different pieces (objects) in the shebeen that have hooks

(define room-pieces

(list

"dcs_shebeen"

"dcs_roof"

"dcs_roof_pillars"

"dcs_walls"

"dcs_front_wall"

"dcs_back_wall"

"dcs_boxing_photo"

"dcs_soccer"

"dcs_jazz"

"dcs_left_wall"

"dcs_window"

"dcs_right_wall"

"dcs_wall_pillars"

"dcs_floor"

"dcs_crates"

"dcs_crates_left"

"dcs_crate_1"

"dcs_crate_2"

"dcs_crate_3"

"dcs_crate_4"

"dcs_crate_tables"

"dcs_crate_table_1"



"dcs_cups"
"dcs_cup_1"
"dcs_cup_2"
"dcs_cup_3"
"dcs_bucket_crate_1"
"dcs_bucket_1"
"dcs_bucket_2"
"dcs_bucket_3"
"dcs_crate_table_2"
"dcs_cup_4"
"dcs_cup_5"
"dcs_cup_6"
"dcs_cup_7"
"dcs_candle_1"
"dcs_cup_8"
"dcs_radio"
"dcs_aerial"
"dcs_handle"
"dcs_button_top"
"dcs_button_bottom"
"dcs_konka"
"dcs_table_ground"
"dcs_bucket_4"
"dcs_candle_2"
"dcs_cup_9"
"dcs_chair_1"
"dcs_chair_2"
"dcs_chair_4"
"dcs_chair_3"
"dcs_extra_heap"
"dcs_crate_5"
"dcs_crate_6"
"dcs_crate_7"
"dcs_crate_8"
"dcs_table_up"
"dcs_bed_top"
"dcs_sheets_1"



"dcs_cushion_1"

"dcs_frame_1"

"dcs_bed_bottom"

"dcs_sheets_2"

"dcs_cushion_2"

"dcs_frame_2"

)

)

Bibliography

- [1] Penny S. Virtual reality as the End of the Enlightenment Project. *Ideologies of Technology Symposium*, New York, 1992.
http://www-art.cfa.cmu.edu/penny/texts/VR_Dia.html
- [2] Johnson A, Moher T, Ohlsson S, and Gillingham M. The Round Earth project - collaborative VR for conceptual learning. *IEEE Computer Graphics and Applications*, 19(6):60-69, November/December 1999.
- [3] Anstey J, and Pape D. Animation in the CAVE, *Animation world magazine*, Issue3.1, April 1998.
- [4] Anstey J, Pape D, and Sandin D. Building a VR narrative, *In Proceedings of SPIE, Stereoscopic Displays and Virtual Reality Systems VII (The Engineering Reality of Virtual Reality 2000)*, 3957, San Jose, CA, January 28, 2000.
- [5] Cooper M, and Benjamin I. Dramatic interaction in virtual worlds. *In Proceedings of 2nd Conference of the UK Virtual Reality Special Interest Group (UK-VRSIG)*, Silicon Graphics Reality Centre, Theale, Reading, UK, December 1, 1994.
- [6] Elliott C. Hunting for the Holy Grail with "Emotionally Intelligent" virtual actors. *Sigart Bulletin*, 9 (1):20-28, 1998.
- [7] Fischnaller F. The Goddess of the Millenium. *The Berliner Festspiele Exhibition: "Seven Hills: Images and Signs of the 21st Century"*, Berlin, Germany, May 14-29, 2000.
- [8] Funge J, Tu X, and Terzopoulos D. Cognitive Modeling: Knowledge, Reasoning and Planning for Intelligent Characters. *SIGGRAPH '99*, Los Angeles, CA, August 8-13, 1999.
- [9] Hirsh H. Trends & Controversies: Interactive fiction. *IEEE Intelligent Systems: Trends and Controversies*, "Interactive Fiction: The Story is Just Beginning", November, 1998.

- [10] Isdale J. *What is virtual reality?*. www.isj
- [11] Jackson C, and Lalioti V. Virtual cultural identities. *CHI-SA*, Human Computer Interaction in South Africa, University of Pretoria Conference Center, Pretoria, South Africa, May 8-10, 2000.
- [12] Johnson A, Roussos M, Leigh J, Vasilakis C, Barnes C, and Moher T. The NICE project: learning together in a virtual world. *Proceedings of VRAIS '98*, 1998 IEEE Virtual Reality Annual International Symposium, Sheraton Colony Square, Atlanta, GA, USA, May 14-18, 1998.
www.evl.uic.edu/tile/NICE/NICE/PAPERS/VRAIS/vrais98.2.html
- [13] Johnson WL, and Rickel J. Steve: An animated pedagogical agent for procedural training in virtual environments. *SigArt Bulletin*, 8 (1-4), 1997.
- [14] Kelso MT, and Weyhrauch P. Dramatic Presence. *Presence: Teleoperators and Virtual Environments*, 2(1):1-15, 1992.
- [15] Kuzmanovic M. Immersive storytelling: Hypermedia in VR. *DAC '99*, Digital Arts and Culture conference, Atlanta, Georgia, USA, October 28-30, 1999.
- [16] Pape D, Imai T, Anstey J, Roussou M, and DeFanti T. XP: An authoring system for immersive art exhibitions, *In Proceedings of VSMM'98*, Fourth International Conference on Virtual Systems and Multimedia, Gifu, Japan, Nov 18-20, 1998.
- [17] Pausch R. Disney's Aladdin: First steps toward storytelling in virtual reality. *In Proceedings of SIGGRAPH'96*, New Orleans, LA, August 4-9, 1996, p. 193-203.
www.siggraph.org/conferences/siggraph96/core/conference/papers/t130.1.html.
- [18] Perlin K, and Goldberg A. Improv: A system for scripting interactive actors in virtual worlds. *In Proceedings of SIGGRAPH'96*, New Orleans, LA, August 4-9, 1996.
- [19] Perlin K, and Goldberg A. *Improvisational animation*. White paper,
www.mrl.nyu.edu/improv/whitepaper/index.html.
- [20] Porter T. Creating lifelike characters in Toy Story. *SigArt Bulletin*, 8(1-4):10-14, 1997.
- [21] Talin. Real interactivity in interactive entertainment. *Computer Graphics*, 28 (2):97-99, May 1994.



- [22] Talin. Sequence vs. Interactivity. *Journal*. <http://www.sylvantech.com/~talin/sequence.html>
- [23] ICE and EVL, The Narrative Immersive Constructionist/Collaborative Environments project. <http://www.evl.uic.edu/tile/NICE/NICE/intro.html>.
- [24] Crawford C. *Erazzmatazz*. <http://www.erasmatazz.com>
- [25] The MIT Media Lab. *GULLIVR (Graphical User Learning Landscapes In VR)*. October, 1996. www.evl.uic.edu/tile/NICE/NICE/gullivr.html
- [26] The MIT Media Lab. *KidsRoom: Action Recognition in an Interactive Story Environment*. October 1996. <http://vismod.www.media.mit.edu/vismod/demos/kidsroom>
- [27] Costa M, and Feijo B. Agents with emotions in behavioral animation, *Computers & Graphics*, 20(3):377-384, 1996.
- [28] Economou D, Mitchell WL, and Boyle T. Requirements elicitation for virtual actors in collaborative learning environments, *Computers & Education*, 34(3-4):225-239, April, 2000.
- [29] Edgar S. The artificial intelligensia and virtual worlds. *Computers & Education*, 27(2): 27-31, June, 1997.
- [30] Laurillard D. Multimedia and the learner's experience of narrative. *Computers & Education*, 31(2):229-242, September, 1998.
- [31] Thomas B. *The Roleplay Workshop for kids*. <http://www.roleplay-workshop.com/forkids.html>
- [32] Thomas B. *The Roleplay Workshop for parents*. <http://www.roleplay-workshop.com/forparents.html>
- [33] Zabel, J., *Web of deceit: Joe's Amazing Montage*. <http://members.tripod.com/~amazingmontage/realism.html>
- [34] Thom. *CafeTecnos: Psst, wanna do a phrontisterion??*. <http://www.indiana.edu/~slizzard/phrontisterion/>
- [35] ORI. *Research areas at Oregon Research Institute: Education & virtual reality*. <http://www.ori.org/educationvr.html>

- [36] Basso M, and Bonotto C. Forum of ideas: is it possible to use cultural heritage to change attitudes about mathematics and its teaching? Exemplifications from an activity in primary schools. *CIEAEM51*, The 51st conference of the International Commission for the Study and Improvement of Mathematics Education, The Mathematics Centre, University College Chichester, Chichester, UK, July 21-26, 1999.
<http://www.chi.ac.uk/conferences/CIEAEM51/papers/forum.html>
- [37] Insley JA, Sandin DJ, and DeFanti TA. Using video to create avatars in virtual reality. *Visual Proceedings of the 1997 SIGGRAPH Conference*, Los Angeles, CA, August, 1997, pp 128.
<http://www.evl.uic.edu/insley/VideoAvatars/VAsketch.html>
- [38] Otto G. Procedural animation. Seminar notes, *Graphics Short Course IV: 3D Animation Techniques*, September 19, 2000.
http://www.psu.edu/dept/cac/viz/sem_notes/animation/index.html
- [39] Johnson A, Moher T, Leigh J, and Lin Y. Quickworlds: Teacher driven VR worlds in an Elementary school curriculum. *SIGGRAPH 2000 Educator's Program*, New Orleans, LA, July 23-28, 2000.
- [40] Kuzmanovic M. *Interactive Noir*. <http://www.cwi.nl/~maja/thesis/chapt.html>
- [41] Leymarie F, and Gruber M. Authoring tools for photorealistic visits and tele-use of the European heritage, architecture & urban tissue. *3rd European Digital Cities Conference*, Berlin, Germany, December 1-2, 1997.
<http://www.lems.brown.edu/vision/people/leymarie/Publish/EDC98JA8.html>
- [42] Van der Post L. *The Lost World of the Kalahari*. Penguin Books, UK, 1962.
- [43] Lombard M, Ditton T. At the heart of it all: The concept of presence, Department of Broadcasting, Telecommunications & Mass Media, Temple University, *JCMC*, 3(2), September 1997.
- [44] Rich BW. *The Saturday Scientists: Experimenting with simple animation devices*.
<http://www.west.net/~science/animate.htm>
- [45] Realsoft. *Animation*. <http://www.realsoft.fi/version4/animation.htm>
- [46] Azar B. The VR pricetag: Nausea, dizziness and disorientation. *The APA Monitor*, 27(3):25, 1996.
- [47] Rothbaum BO, Hodges LF, Kooper R, Opdyke D, Williford JS, and North M. Effectiveness of computer-generated (virtual reality) graded exposure in the treatment of acrophobia. *American Journal of Psychiatry*, 152(4):626-628, 1995.

- [48] Nass C, Moon Y, Fogg BJ, Reeves B, and Dryer DC. Can computers be human personalities? *International Journal of Human-Computer Studies*, 43:223-239, 1995.
- [49] Economou D, Mitchell WL, and Boyle T. Requirements elicitation for virtual actors in collaborative learning environments. *Computers & Education*, 34(3-4): 225-239, April/May, 2000.
- [50] Takala T. The electronic garden. *SIGGRAPH '97 Exhibition*, Los Angeles Convention Centre, Los Angeles, USA, August 3-8, 1997.
- [51] Fabricators. *The Multi Mega Book*. <http://www.evl.uic.edu/EVL/NEWS/MMB/index.html>
- [52] Fabricators. *Virtual reality and Interactivity*. <http://www.fabricat.com/VRI99WEB/ENG/sec3.html>
- [53] Robertson B. Motion Capture Meets 3D Animation, in Ringo T. *On the Cutting Edge of Technology*, Sams Publishing, Carmel, USA, 1993, p. 1-14.
- [54] Murray JH. *CharacterMaker 4.1*. http://caes.mit.edu/research/character_maker/
- [55] Bush MD, and Terry RM. *Technology-enhanced language learning: Multiple, Mixed, Malleable Media*, National Textbook Company, Lincolnwood, Illinois, 1997, p. 48.
- [56] Maes P, Darrell T, and Blumberg B. The Alive System: Full Body Interaction with Autonomous Agents. *In Proceedings of Computer Animation '95 Conference*, Geneva, Switzerland, April 17-21, 1995, p. 11-18.
- [57] Leymarie F, and Gruber M. *Applications of Virtual Reality: CyberMonument and CyberCity*, 3rd European Digital Cities Conference, Berlin, Germany, December 1-2, 1997. (already there)
- [58] Debord G. *Society of the Spectacle*. 1967.
<http://www.marxists.org/reference/subject/philosophy/works/fr/debord.htm>
- [59] ORI. *Research areas at Oregon Research Institute: Virtual Reality*. <http://www.ori.org/educationvr.html>.
- [60] Burkin V, Göbel M, Hasenbrink F, Klimenko S, Nikitin I, and Tramberend H. *Virtual Planetarium in CyberStage*.
<http://viswiz.gmd.de/~nikitin/stars/starsCB.html>
- [61] Elliot, TS. *Four Quartets: Little Gidding*. Faber And Faber, UK, 2000.

- [62] FHH, Foundation of the Hellenic World. *3D modeling and animation projects: Miletus*.
<http://www.fhw.gr/fhw/en/projects/3dvr/miletus/index.html>.
- [63] Adams E. *Three problems for interactive storytellers*. Gamasutra Designers' Notebook, December 1999.
http://www.gamasutra.com/features/designers_notebook/19991299.htm
- [64] Juul J. A clash between Game and Narrative. *Digital Arts and Culture*, Bergen, November 1998.
- [65] Genettes G. *Narrative discourse*. Cornell University Press, Ithaca, NY, USA, 1980, p. 33.
- [66] Locatis, Ullmer, Carr, Banvard, Le, Lo, and Williamson. Authoring system reassessed. *Educational Technology Research and Development*, 40(2):77-82, 1992.
- [67] Tramberend H. AVANGO: A distributed virtual reality framework. *In Proceedings of the IEEE Virtual Reality '99*, Houston, Texas, March 13-17, 1999.
- [68] Lynch B. *Cato Manor development project*. <http://www.cmda.org.za>
- [69] Hu J. *Spatial ability and hypermedia authoring: The attitudes of novice hypermedia developers and the characteristics of their concept maps and authoring projects*.
<http://nt-hsta.hsc.wvu.edu/health/personal/AERAProposal.html>
- [70] Youngblut C. *Educational Uses of virtual reality technology*. IDA Document D-2128, Alexandria, VA: Institute for Defense Analyses, 1998.
- [71] Cronin P. *Report on the applications of virtual reality technology to education*. HCRC, University of Edinburgh, February 1997.
<http://www.cogsci.ed.ac.uk/~paulus/Work/Vranded/vr.htm>
- [72] Dickey MD. *3D virtual worlds and learning: an analysis of the impact of design affordances and limitations in Active Worlds, Blaxxun Interactive, and Onlive! Traveler; and a study of the implementation of Active Worlds for formal and informal education*. Dissertation, Graduate School of the Ohio State University, 1999.
- [73] Best K. *The idiot's guide to virtual world design*. Little Star Press, Seattle, WA, 1994.
- [74] Vincelli F. From imagination to virtual reality: the future of clinical psychology. *CyberPsychology & Behaviour*, 2(3), 1999.



- [75] Shawver DM. Virtual actors and avatars in a virtual environment. *In Proceedings of the IEEE 1997 Virtual Reality Annual International Symposium*, Albuquerque, NM, March 1-5, 1997.
- [76] Ellis RE. Human engineering in virtual environments. *Virtual Reality World '95*, Stuttgart, Germany, 1995, p. 295-302.
- [77] Clinger W, and Rees J. *Revised report on the algorithmic language Scheme*, November 1991.
- [78] Anstey J, Pape D, and Sandin D. The Thing Growing: Autonomous characters in virtual reality interactive fiction. *Proceedings of IEEE Virtual Reality 2000*, New Brunswick, NJ, March 18-22, 2000, p. 71-78.
- [79] Laurel B, Strickland R, and Tow R. Placeholder: Landscape and narrative in virtual environments. *ACM Computer Graphics*, 28(2), May 1994.
- [80] Abdelmawla S. Virtual environment for structural education. *ACADIA '98*, Quebec City, Quebec, Canada, October 22-25, 1998.
- [81] Kallman M, and Thalmann D. Direct 3D interaction with smart objects. *VRST'99*, Symposium on Virtual Reality Software and Technology, University College, London, 1999, p. 124-130.
- [82] Bowman DA, Johnson DB, and Hodges LF. Testbed evaluation of virtual environment interaction techniques. *VRST '99*, Symposium on Virtual Reality Software and Technology, University College, London, 1999.
- [83] Codognet P. *Declarative behaviors for virtual creatures*. *ICAT '99*, International conference on Auditory Display, Waseda University International Conference Center, Tokyo, Japan, December 16-18, 1999, p. 208-214.
- [84] Dai P, Eckel G, Göbel M, Hasenbrink F, Lalioti V, Lechner U, Strassner J, Tramberend H, and Wesche G. Virtual spaces: VR projection system technologies and applications. *Eurographics '97*, European computer graphics conference, Budapest, Hungary, September 4-8, 1997.
- [85] Ware C, Arthur K, and Booth KS. Fish Tank Virtual Reality. *INTERCHI '93*, Proceedings of the international conference on human computer interaction, Amsterdam, the Netherlands, April 24-29, 1993, p. 37-42
- [86] Smith AGM. *HandyMan – A multiuser puppeteering system for computer graphics motion control*. M.Sc. thesis, School of Computer Science, Carleton University, Ontario, Ottawa, 1991.
<http://home.achilles.net/~agmsmith/handyman/Thesis.html#Animation>



[87] Dyer S, Martin J, and Zulauf J. *Motion capture*. December 12, 1995.

http://reality.sgi.com/jam_sb/mocap/MoCapWP_v2.0.html

[89] Herndon K, van Dam A, Gleicher M. The Challenges of 3D Interaction. *SIGCHI Bulletin*, 26(4), October 1994, p. 36-43.

[90] Mateus M. *An Oz-Centric Review of Interactive Drama and Believable Agents*. Technical Report CMU-CS-97-156, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, June, 1997.

[91] Macedonia M. Entertainment Computing: Using Technology and Innovation to Simulate Daily Life. *Computer*, 33(4), April 2000, p. 110-112.

[92] Greeff M, Lalioti V. Interactive Storytelling with Virtual Identities. *IPT/EGVE 2001*, 5th International Projection Technology Workshop and 7th Eurographics Workshop on Virtual Environments, Stuttgart, Germany, May 16-18, 2001.

[93] Greeff M, Lalioti V. Authoring Interactive stories using Virtual Identities. *EuroImage 2001*, International Conference on Augmented, Virtual Environments and 3D Imaging, Mykonos, Greece, May 30-June 1, 2001.

[94] Greeff M, Lalioti V. Interactive Cultural Experiences using Virtual Identities. *ICHIM 2001*, International Conference on Cultural Heritage and Technologies in the Third Millennium, Milan, Italy, September 3-7, 2001.

[95] Göbel M, Barrass S, Eccles J, Eckel G, Hasenbrink F, Lechner U, Mostafawy S, Toth S, Tramberend H, Unnützer, P. Digital Storytelling Creating Interactive Illusions with AVOCADO, *ICAT '99*, 9th International Conferences on Artificial Reality and Telexistence, Waseda University International Conference Center, Tokyo, Japan, December 16-18, 1999.

[96] Weinbren G. An interactive Cinema, Time, Tense and some Models. *New Observations*, 71, 1989.

[97] Nolan DJ. *Educators in MOOKti: A Polysynchronous Collaborative Virtual Learning Environment*. Thesis Proposal for the Degree of Doctor of Philosophy, Ontario Institute for Studies in Education, University of Toronto, Canada, 1998. <http://noisy.oise.utoronto.ca/jason/mookti-res.html>

[98] Huhtamo E. Commentaries on Metacommentaries on Interactivity. *Cultural Diversity in the Global Village*, The Third International Symposium on Electronic Art, Sydney, Australia, November 9-13, 1992.

Translated quote from:

Moeglin P. *Les transinteractifs*. Derrick de Kerckhove and Christian Sevette (eds), Espace SNVB international, Paris, 1990, p 106.

[99] Craig GJ. *Human Development*. Prentice Hall, New Jersey, USA, 1996.

[100] De la Rey C, Duncan N, Shefer T, Van Niekerk A. *Contemporary issues in Human Development: A South African Focus*. International Thomson Publishing (Southern Africa) (Pty) Ltd, South Africa, 1997.