Università degli studi di Napoli
Parthenope

Corso di laurea in Informatica

Progetto Realtà Virtuale

Prof. Francesco Camastra

Humanoid Woman

di Tiziana Ercolani

Sommario

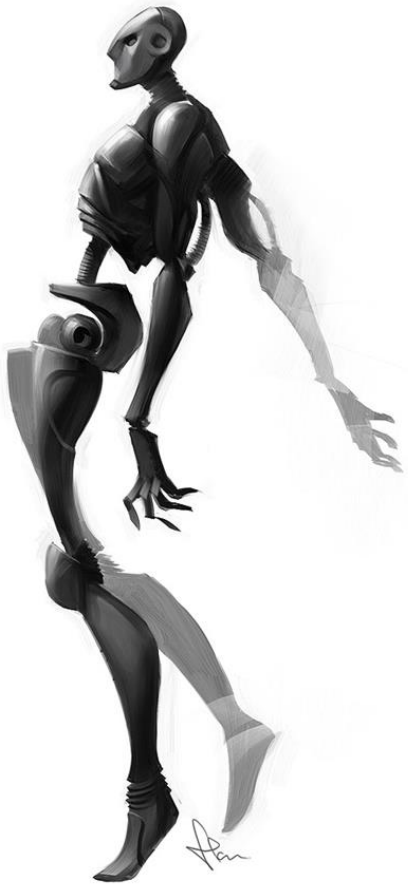
Introduzione

Il progetto Humanoid Woman prevede la realizzazione virtuale di un umanoide donna che effettua la camminata.

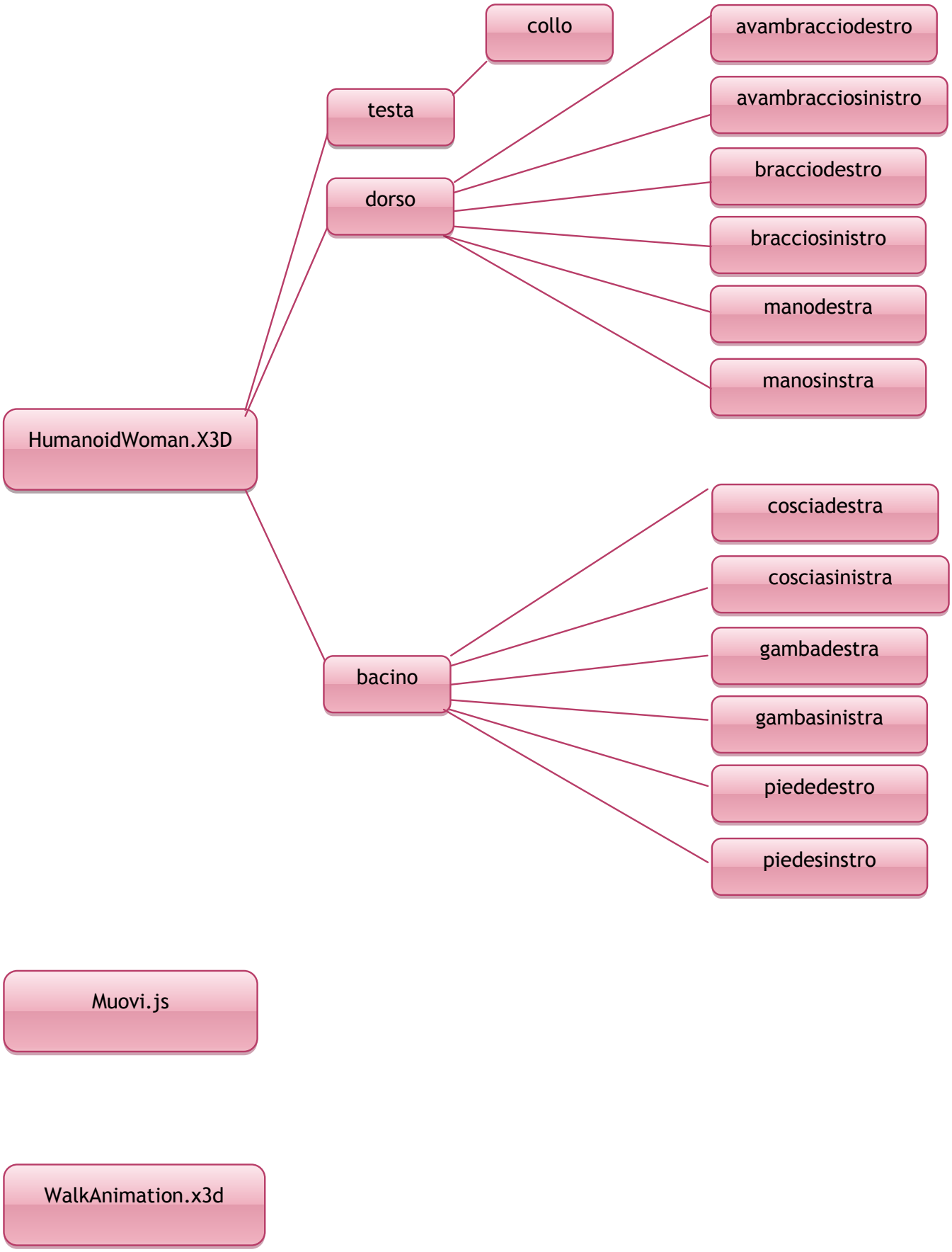
La scelta di questo progetto è stata effettuata per accurare uno studio più approfondito sulle animazioni dell'essere umano ,in quanto sono famose per essere molto complesse.

Per effettuare questo progetto ho lavorato con Blender per quanto riguarda la modellazione statica del soggetto, mentre ho implementato le animazione con X3D e utilizzato le Script per richiamare alcune animazioni.

I player utilizzati sono Instantplayer ed Octoga.



Tassonomia del Progetto

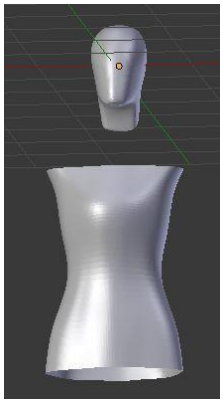


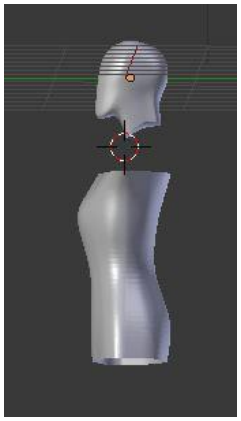
Modellazione Umanoide

Come ho già anticipato la modellazione è stata effettuata in Blender, un potente software per la costruzione di oggetti 3D ed esportare/importare codice X3D.

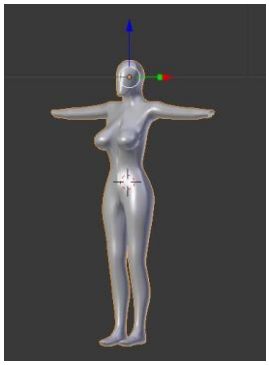
Grazie alle operazioni di Mesh ed Estrusioni sono riuscita a modellare un corpo femminile partendo da un comune cubo 3D.

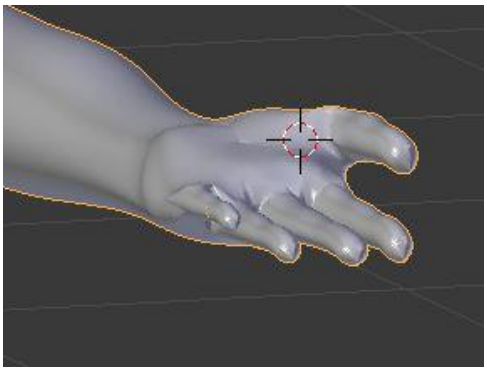
Ho quindi prima modellato la testa ed il busto :



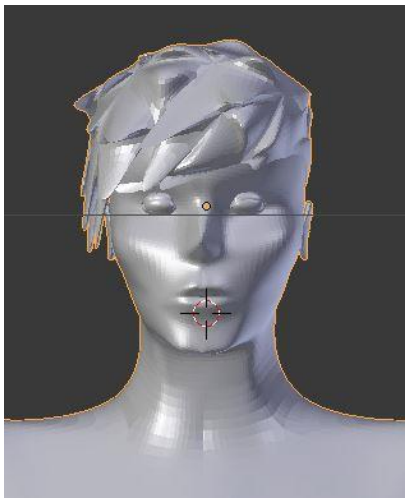


Successivamente ho costruito le gambe, le braccia e le mani particolarmente curate nel dettaglio:





Ho poi modellato il viso e costruito i capelli partendo da una icosfera:



Ho costruito infine un oggetto “maglietta” , da applicare sul corpo della donna, partendo sempre da un semplice cubo:

L’ultimo passo è stato quello delle texture ho scelto infatti di non applicare un'unica texture per tutto l’umanoide , ma di applicare la texture della pelle sul corpo, un’immagine di tessuto per la t-shirt e infine ho semplicemente scelto il colore nero per i pantaloni e le scarpe e il colore marrone per i capelli. L’implementazione delle texture o della colorazione l’ho applicata tramite il nodo **Matereal**:

<Material DEF="MA_Material_012"

diffuseColor="0.006 0.006 0.006”

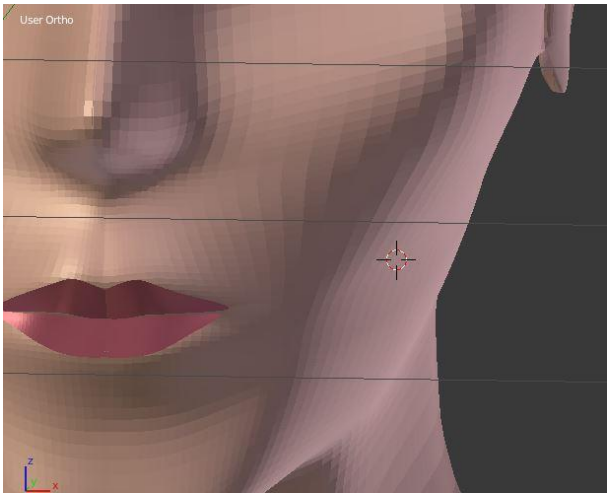
specularColor="0.070 0.070 0.070"

emissiveColor="0.000 0.000 0.000"

ambientIntensity="0.333"

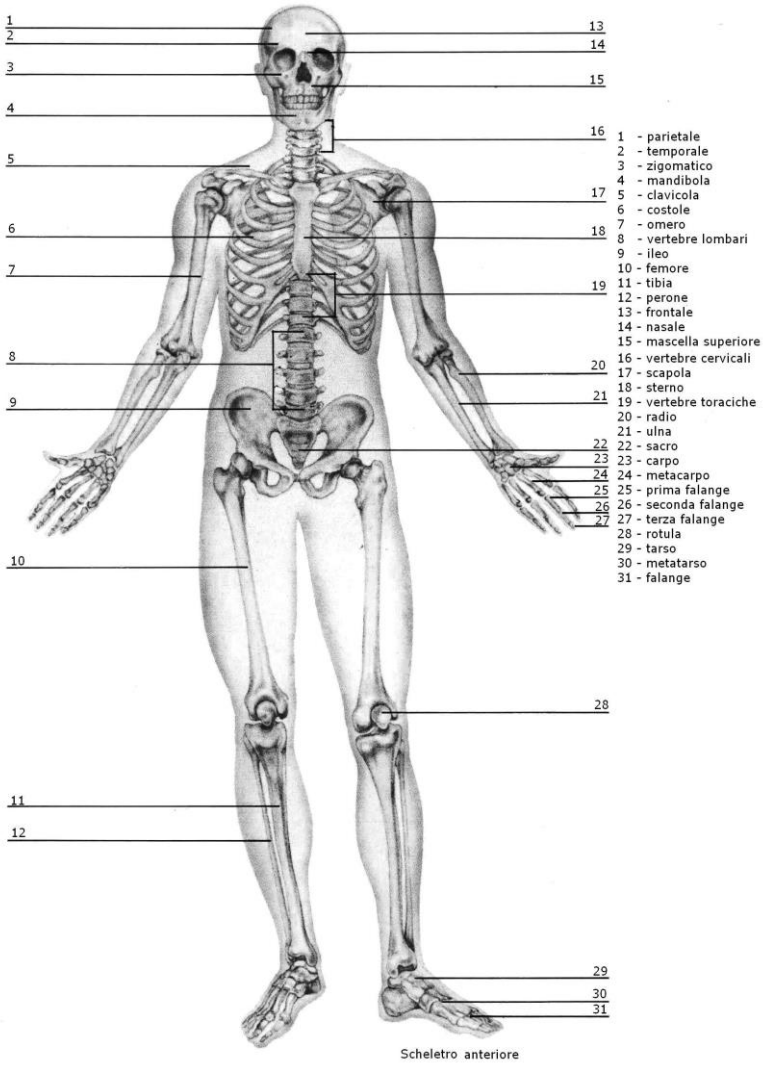
shininess="0.098"

transparency="0.0"



Implementazione delle giunture

L’implementazione delle giunture dell’essere umano è un lavoro complesso in quanto le articolazioni umani sono tante e svolgono diversi movimenti.



Scheletro anteriore

Per consentire al mio umanoide di camminare ho dovuto implementare solo alcuni di essi quindi ho escluso tutte le articolazioni delle dita ,dei piedi, della mascella e i movimenti dei viso che spero di approfondire in futuro.

Le **Joint** sono state implementate in X3D. Grazie alla libreria **H-anim** che consente di utilizzare il nodo **HanimHumanoid** .

Il nodo **HAnimHumanoid** viene usato per:

- memorizzare i riferimenti a giunture, segmenti e viste
- fare da contenitore per l'intero umanoide
- fornire un modo appropriato per muovere l'umanoide nel suo ambiente
- memorizzare i dati leggibili da un umano

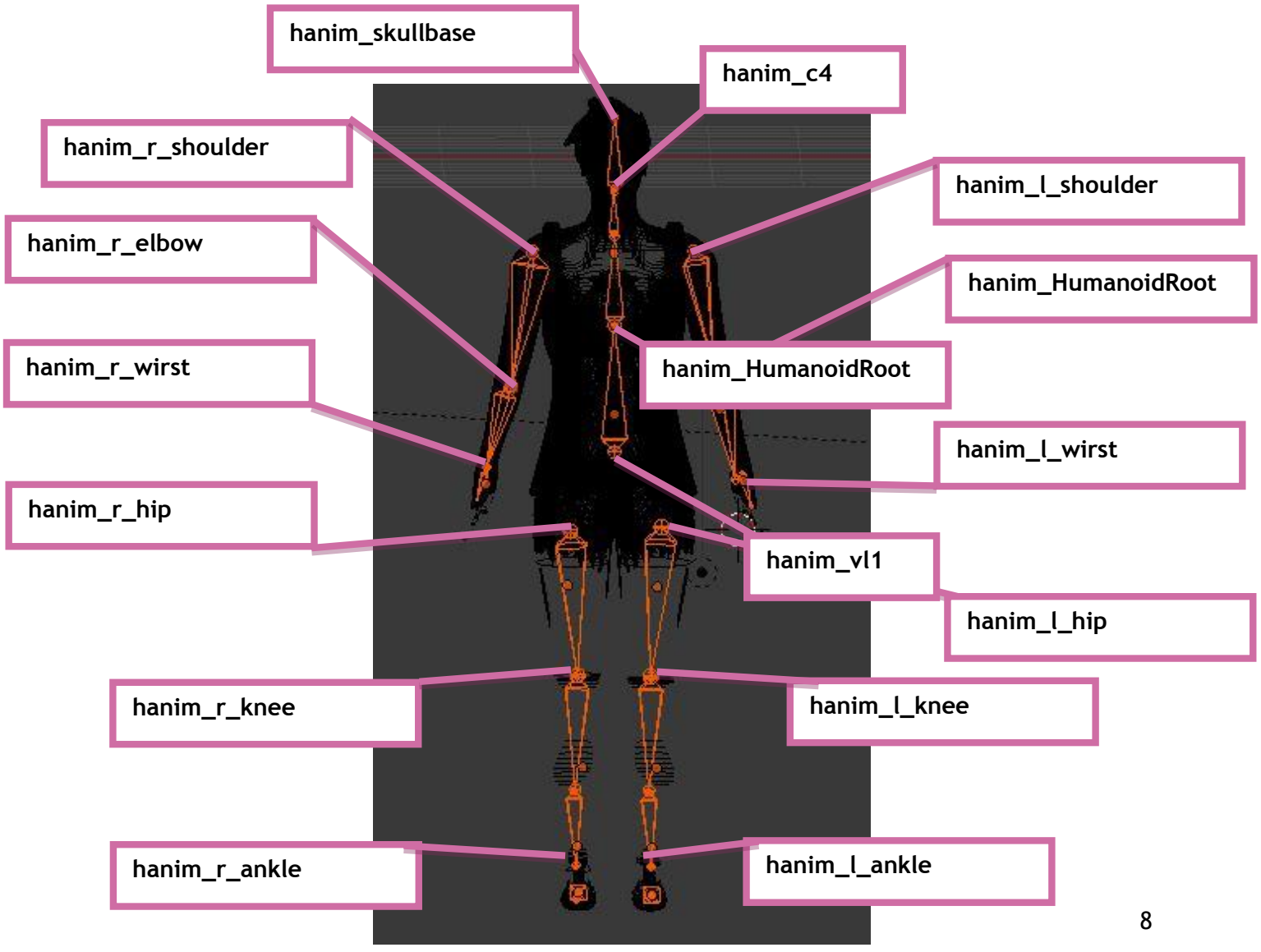
HAnimHumanoid contiene nodi **HAnimJoint, HAnimSegment, HAnimSite**, Coordinate/CoordinateDouble, Normal, Viewpoint e skin.

Ogni giuntura del corpo è rappresentata da un nodo HAnimJoint.

HAnimJoint può essere solo figlio di un altro **nodo HAnimJoint**, oppure il campo skeleton di **HAnimHumanoid**.

Ogni segmento del corpo viene memorizzato in un nodo **HAnimSegment**.

HAnimSegment contiene nodi Coordinate/CoordinateDouble, HAnimDisplacer e nodi figli.



Ho definito ogni Joint nel seguente modo:

```
HAnimHumanoid DEF='Humanoid' name='Humanoid' info=''>
  <HAnimJoint DEF='hanim_HumanoidRoot' name='HumanoidRoot' center='0 0 0' containerField='skeleton' >
    <HAnimJoint DEF='hanim_sacroiliac' name='sacroiliac' center='-0.054394 -2.865425 0.211846'>
      <HAnimSegment DEF='hanim_pelvis' name='pelvis'>
        <Transform translation='-0.054394 -2.865425 0.211846'>

          <Shape DEF='Bacino'>
          <Appearance>
          <Material diffuseColor='1.0 1.0 0.0'/>
          </Appearance>
          //Coordinate dell oggetto che corrisponde al bacino
          </Shape>

        </Transform>
```

Implementando in ogni Joint la parte del corpo che sarà legata a quell tipo di giunzione.

Effettuando quest'istruzione per ogni Joint ho costruito il mio umanoide. Infine ho dichiarato tutti i Joint ed i Segment utilizzati :

```
<HAnimJoint USE='hanim_r_wrist' containerField='joints'/>
<HAnimJoint USE='hanim_r_elbow' containerField='joints'/>
<HAnimJoint USE='hanim_r_shoulder' containerField='joints'/>
<HAnimJoint USE='hanim_l_wrist' containerField='joints'/>
<HAnimJoint USE='hanim_l_elbow' containerField='joints'/>
<HAnimJoint USE='hanim_l_shoulder' containerField='joints'/>
<HAnimJoint USE='hanim_skullbase' containerField='joints'/>
<HAnimJoint USE='hanim_vc4' containerField='joints'/>
<HAnimJoint USE='hanim_vl1' containerField='joints'/>
<HAnimJoint USE='hanim_r_ankle' containerField='joints'/>
<HAnimJoint USE='hanim_r_knee' containerField='joints'/>
<HAnimJoint USE='hanim_r_hip' containerField='joints'/>
<HAnimJoint USE='hanim_l_midtarsal' containerField='joints'/>
<HAnimJoint USE='hanim_l_ankle' containerField='joints'/>
<HAnimJoint USE='hanim_l_knee' containerField='joints'/>
<HAnimJoint USE='hanim_l_hip' containerField='joints'/>
<HAnimJoint USE='hanim_HumanoidRoot' containerField='joints'/>
```

<!-- top-level segment references -->

<HAnimSegment USE='hanim_sacrum' containerField='segments'/>

<HAnimSegment USE='hanim_l_thigh' containerField='segments'/>

<HAnimSegment USE='hanim_l_calf' containerField='segments'/>

<HAnimSegment USE='hanim_l_hindfoot' containerField='segments'/>

<HAnimSegment USE='hanim_c4' containerField='segments'/>

<HAnimSegment USE='hanim_r_thigh' containerField='segments'/>

<HAnimSegment USE='hanim_r_calf' containerField='segments'/>

<HAnimSegment USE='hanim_r_hindfoot' containerField='segments'/>

<HAnimSegment USE='hanim_r_middistal' containerField='segments'/>

<HAnimSegment USE='hanim_l1' containerField='segments'/>

<HAnimSegment USE='hanim_skull' containerField='segments'/>

<HAnimSegment USE='hanim_l_upperarm' containerField='segments'/>

<HAnimSegment USE='hanim_l_forearm' containerField='segments'/>

<HAnimSegment USE='hanim_l_hand' containerField='segments'/>

<HAnimSegment USE='hanim_r_upperarm' containerField='segments'/>

<HAnimSegment USE='hanim_r_forearm' containerField='segments'/>

<HAnimSegment USE='hanim_r_hand' containerField='segments'/>

</HAnimHumanoid>

Animazione

Una volta sviluppate le Joint, del corpo della mia umanoide, è stato possibile gestire le animazioni.

Il primo passo è stato quello di dichiarare tutti i tipi InputType o OutputType che vengono usati nell'animazione:

```
<ExternProtoDeclare name='WalkAnimation' url='"WalkAnimation.x3d"'>
    <field accessType='inputOutput' name='cycleInterval' type='SFTime'/>
    <field accessType='inputOutput' name='enabled' type='SFBool'/>
    <field accessType='inputOutput' name='loop' type='SFBool'/>
    <field accessType='inputOutput' name='startTime' type='SFTime'/>
    <field accessType='inputOutput' name='stopTime' type='SFTime'/>
    <field accessType='outputOnly' name='fraction_changed' type='SFFloat'/>
    <field accessType='outputOnly' name='HumanoidRoot_translation_changed' type='SFVec3f'/>
    <field accessType='outputOnly' name='HumanoidRoot_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='l_hip_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='l_knee_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='l_ankle_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='l_midtarsal_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='r_hip_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='r_knee_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='r_ankle_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='r_midtarsal_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='vl5_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='skullbase_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='l_shoulder_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='l_elbow_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='l_wrist_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='r_shoulder_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='r_elbow_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='r_wrist_rotation_changed' type='SFRotation'/>
    <field accessType='outputOnly' name='isActive' type='SFBool'/>
  </ExternProtoDeclare>
```

Successivamente collego le animazioni alle Joint a cui di riferiscono tramite il nodo RUOTE.

ROUTE connette dei campi di nodi diversi per permettere la trasmissione di eventi:

```
      <!-- Animation ROUTEs -->


   <ROUTE fromField='HumanoidRoot_translation_changed' fromNode='ANIMATOR'

toField='set_translation' toNode='hanim_HumanoidRoot'/>

   <ROUTE fromField='HumanoidRoot_rotation_changed' fromNode='ANIMATOR'

toField='set_rotation' toNode='hanim_HumanoidRoot'/>

   <ROUTE fromField='l_hip_rotation_changed' fromNode='ANIMATOR' toField='set_rotation'

toNode='hanim_l_hip'/>

   <ROUTE fromField='l_knee_rotation_changed' fromNode='ANIMATOR' toField='set_rotation'

toNode='hanim_l_knee'/>

   <ROUTE fromField='l_ankle_rotation_changed' fromNode='ANIMATOR' toField='set_rotation'

toNode='hanim_l_ankle'/>

   <ROUTE fromField='l_midtarsal_rotation_changed' fromNode='ANIMATOR'

toField='set_rotation' toNode='hanim_l_midtarsal'/>

   <ROUTE fromField='r_hip_rotation_changed' fromNode='ANIMATOR' toField='set_rotation'

toNode='hanim_r_hip'/>

   <ROUTE fromField='r_knee_rotation_changed' fromNode='ANIMATOR' toField='set_rotation'

toNode='hanim_r_knee'/>

   <ROUTE fromField='r_ankle_rotation_changed' fromNode='ANIMATOR' toField='set_rotation'


toNode='hanim_r_ankle'/>

   <ROUTE fromField='r_midtarsal_rotation_changed' fromNode='ANIMATOR'

toField='set_rotation' toNode='hanim_r_midtarsal'/>

   <ROUTE fromField='vl5_rotation_changed' fromNode='ANIMATOR' toField='set_rotation'

toNode='hanim_vl5'/>

   <ROUTE fromField='skullbase_rotation_changed' fromNode='ANIMATOR'

toField='set_rotation' toNode='hanim_skullbase'/>
```

Ho utilizzato delle script per gestire nel miglior modo i dati in input delle animazioni e poter ottenere le coordinate di particolari field.

Il file “Muovi.js” non farà altro che ricavare le coordinate di ogni joint e restituirle come output per le animazioni.

```
<Script DEF='ENGINE' directOutput='true' url='"Muovi.js"'>
  <field accessType='inputOnly' name='update' type='SFRotation'/>
  <field accessType='initializeOnly' name='humanoid' type='SFNode'>
    <HAnimHumanoid USE='Humanoid'/>
  </field>
  <field accessType='initializeOnly' name='coordList' type='MFVec3f'/>
  <field accessType='initializeOnly' name='joint' type='SFNode' value='NULL'/>
  <field accessType='initializeOnly' name='translation' type='SFVec3f' value='0 0 0'/>
  <field accessType='initializeOnly' name='rotation' type='SFRotation' value='1 0 0
0'/>
  <field accessType='initializeOnly' name='scale' type='SFVec3f' value='1 1 1'/>
</Script>
```

In questo modo definisco gli input e output utilizzati nella script.

L’animazione della camminata verrà gestita tramite il nodo “OrientationInterpolator” che generata una serie di valori di rotazione.

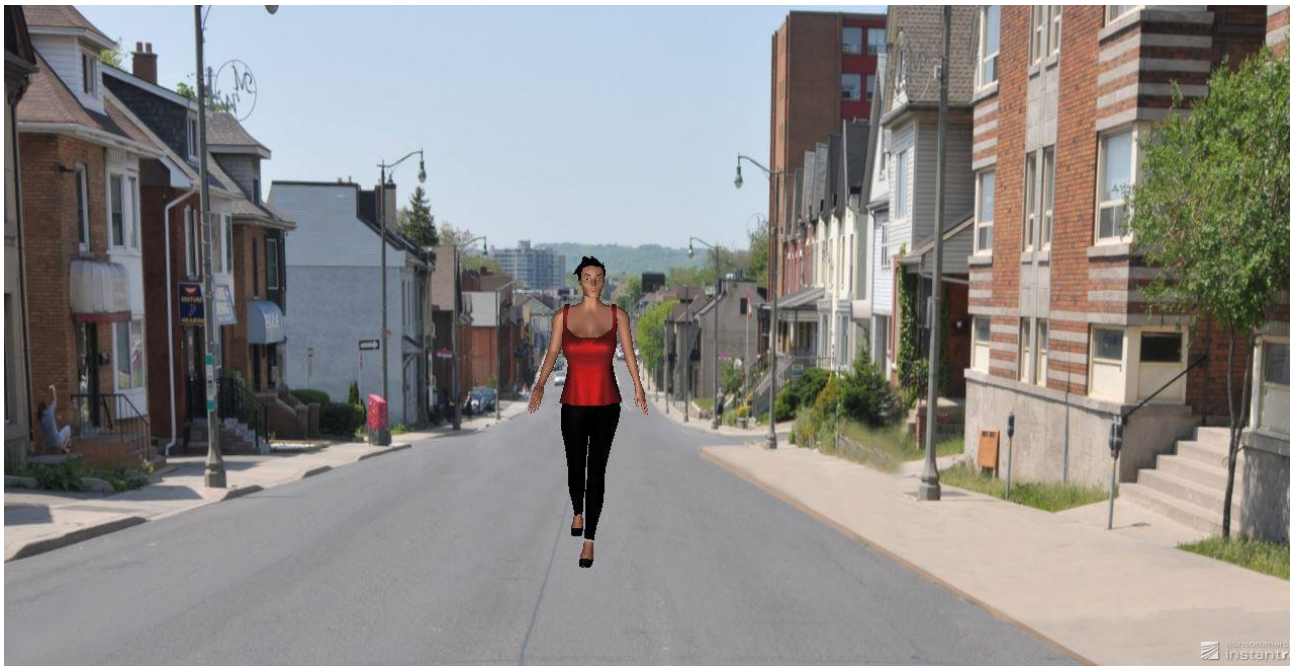
Il risultato viene inviato (tramite ROUTE) all’attributo 'rotation' di un nodo <Transform>.

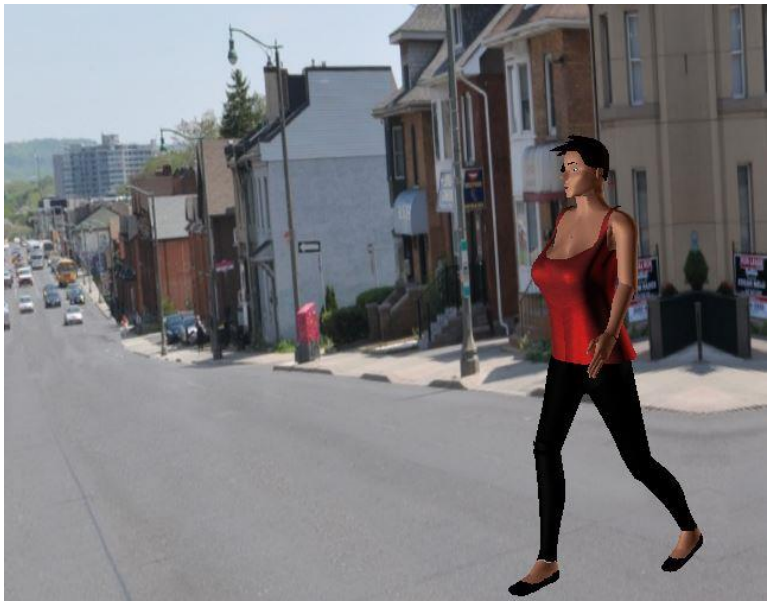
Un esempio di come ho utilizzato il nodo è il seguente dove gestisco la rotazione del ginocchio sinistro:

```
<OrientationInterpolator DEF='L_KNEE_ANIMATOR' key='0 .2083 .375 0.5 0.6667 0.7917 0.9167 1'
keyValue='1 0 0 0.3226 1 0 0 0.1556 1 0 0 0.08678 1 0 0 0.8751 1 0 0 1.131 1 0 0 0.09961 1 0 0 0.3942 1 0 0
0.3226'>
```

Per ogni giunzione viene implementata la giusta rotazione che sarà poi legata alla frazioni di secondi definita nella TimeSensor:

```
<TimeSensor DEF='TIMER'>
```





Problematiche Riscontrate

Una delle problematiche riscontrate è stata nel lavoro con Blender, in quanto le coordinare definite da Blender sono disposte in modo differente dalle coordinate di X3D.

Le coordinate x,y,z di Blender corrispondono alle coordinate x,z,y in X3D. Questo ha comportato un rallentamento delle implementazioni ed un limite sulla gestione corretta della rotazione.

Inoltre in Blender è possibile costruire un'ossatura dell'umanoide tramite il comando 'Armature' la gestione sembra molto vicina a X3D ma non vi è compatibilità. Infatti esportando il file con la corretta Armature dell'oggetto non risulta essere una vera e propria HanimJoint ma un oggetto qualsiasi.

Suggerisco quindi Blender per gestioni di oggetti, modellazione e texturizzazione, ma lo svantaggio dell'incompatibilità è rilevante.

Sviluppi Futuri

Uno degli sviluppi futuri sarà sicuramente la gestione del saluto e quello dei movimenti facciali, in modo da rendere più verosimile il movimento dell'umanoide in questione.

Quindi provvederò ad implementare l'animazione e le giunture del viso per effettuare: l'apertura della bocca, chiusura e apertura delle palpebre movimento degli occhi e infine espressioni facciali.

Riferimenti

- Wikipedia
- X3D:Extensible 3D Graphics for Web Authors di Don Brutzman Leonard Daly
- Slide Prof. Camastra
- http://www.web3d.org/x3d/content/X3dTooltipsItalian.html
- Tutorial Blender: https://www.youtube.com/user/redbaron85ct