# PERANCANGAN ALAT RIGGING KARAKTER OTOMATIS PADA AUTODESK MAYA STUDI KASUS : PT MATARAM SURYA VISI (MSV)

**Aryanto Yuniawan [1)], Roy [2)], Stevi Ema Wijayanti [3)]**

*[1) 2)] Magister Teknik Informatika STMIK AMIKOM Yogyakarta*
*[3)] Desain/Komunikasi Visual Universitas Kristen Petra Surabaya*
*[1) 2)] Jl. Ring Road Utara Condong Catur Depok Sleman Yogyakarta*
*[3)] Jl. Siwalankerto 121-131 Surabaya*
*Email :aryantoyuniawan@gmail.com,[1)] , roytok@yahoo.com[2)] , steviema.wijayanti@gmail.com[3)]*

**Abstrak**

*Rigging karakter merupakan salah satu proses yang sangat penting pada pembuatan film animasi 3D. Rigging difungsikan untuk menciptakan struktur hirarki dari sebuah karakter yang disebut dengan kerangka. Kerangka tersebut digunakan sebagai kerangka kerja (framework) yang dapat memberikan nyawa terhadap sebuah karakter agar dapat berperilaku seperti karakter yang ada di dunia nyata.*

*Waktu yang dibutuhkan untuk melakukan rigging pada sebuah karakter cukup lama dan rentan akan kesalahan jika dilakukan secara manual. Untuk itu diperlukan sebuah alat dimana alat tersebut menyediakan template kerangka karakter. Dengan adanya alat-alat tambahan tersebut dapat memudahkan pekerjaan animator sehingga dapat mempercepat waktu produksi film dan dapat menekan biaya pembuatan film.*

***Kata kunci :*** *rig character, alat rigging otomatis, film animasi, animasi 3D*

## 1. Pendahuluan

### 1.1 Latar Belakang Masalah

Seiring dengan perkembangan zaman telah banyak sekali bermunculan film-film animasi yang dikemas dalam bentuk animasi 3D. Di Indonesia dewasa ini telah mulai berkembang industri yang membuat film animasi baik menggunakan cara manual atau dengan bantuan perangkat lunak (*software*), walaupun masih jauh dibandingkan dengan negara-negara lain. Film animasi 3D memiliki keunikan dimensinya dan sifat hiburannya, oleh karena itu filrm animasi telah diterima sebagai salah satu media audio visual yang paling populer dan paling digemari.

PT Mataram Surya Visi (MSV) merupakan salah satu badan usaha milik STMIK AMIKOM Yogyakarta yang bergerak di industri film animasi. PT MSV telah cukup banyak menghasilkan karya dan berhasil mendapatkan penghargaan baik tingkat nasional maupun internasional. Saat ini, PT MSV telah menyiapkan film animasi 3D yang berjudul "*Ajisaka : Fire and Ice*" dan film animasi 2D yang berjudul "*Battle of Surabaya*" yang akan segera diluncurkan ke masyarakat.

Proses pembuatan film animasi terutama 3 Dimensi (3D) cukup rumit dan membutuhkan waktu yang lama walaupun telah banyak perangkat lunak (*software*) yang tersedia untuk membantu dan mempermudah pekerjaan animator. Untuk itu, masing-masing industri film terutama film animasi membuat alat (*tool*) sendiri yang disesuaikan dengan kebutuhan untuk mempercepat proses pembuatan film. Salah satu alat (*tool*) yang akan dikembangkan oleh PT MSV adalah sebuah alat untuk rigging karakter otomatis. Dengan adanya alat ini diharapkan dapat mempercepat proses pembuatan film animasi 3D sehingga biaya produksi film dapat ditekan.

### 1.2 Tujuan Penelitian

Penelitian ini bertujuan untuk mengembangkan alat (*tools*) rigging manual yang telah disediakan oleh Autodesk Maya menjadi alat rigging otomatis.

Dengan adanya alat ini diharapkan : dapat memperkecil kesalahan-kesalahan (*human error*) yang dilakukan oleh animator, konsistensi hirarki dari tiap-tiap karakter sehingga animasi karakter lebih mudah, dapat mempercepat waktu produksi film (*pipeline*) sehingga dapat menekan biaya produksi.

### 1.3 Batasan Masalah

Agar tidak menyimpang jauh dari permasalahan, maka penelitian ini memiliki batasan masalah sebagai berikut :

1. Alat yang akan dikembangkan adalah alat untuk otomatisasi rigging karakter yang memiliki struktur kerangka mirip dengan kerangka manusia.
2. Implementasi alat menggunakan bantuan Autodesk Maya.

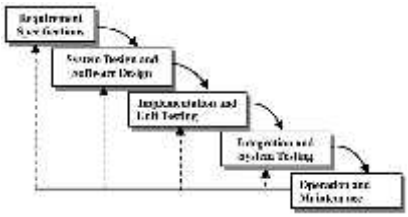
### 1.4 Metodologi Penelitian

Metode penelitian yang digunakan dalam penulisan ini adalah :

1. Studi Pustaka

Penulis merancang alat yang akan dibuat dengan melakukan studi pustaka seperti melalui buku dan jurnal ilmiah ataupun *browsing* melalui *internet*.

2. Studi Lapangan
   a. Metode Pengumpulan Data
      Pada tahap ini penulis mengumpulkan data yang diperlukan dengan cara mendownload, memodifikasi gambar dan scanning gambar yang diambil dari buku dan internet. Data-data lain yang diperlukan dilakukan dengan cara observasi dan wawancara dengan beberapa animator PT MSV yang terlibat langsung dengan produksi film animasi.
   b. Metode Perancangan Alat Rigging Otomatis
      Metodologi yang digunakan adalah menggunakan metodologi SDLC (*Systems Development Life Cycle*) dengan menggunakan model Waterfall.

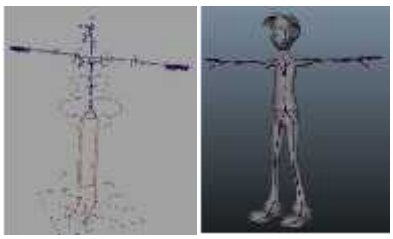


Gambar 1. Waterfall Model

## 1.5 Tinjauan Pustaka

### 1.5.1 Rig

Sebelum membuat sebuah karakter 3D menggunakan Autodesk Maya, harus dikenali terlebih dahulu struktur tulang kerangka dari karakter tersebut. Membuat gerakan karakter yang dibuat menjadi nyata membutuhkan kontrol berdasarkan cara kerja tubuh yang sebenarnya. [1]



Gambar 2. Struktur Tulang Kerangka Karakter

### 1.5.2 Joints

Tulang kerangka dalam aplikasi Maya diciptakan dari obyek yang disebut sendi (*joints*). Sendi ini terikat satu sama lain dalam sebuah hirarki. [2]

Seperti contoh berikut ini : *Joints* untuk kaki karakter.

Gambar 3. Joints for Leg

### 1.5.3 Kinematics

*Kinematics* merupakan metode untuk menggerakan tulang kerangka karakter yang telah dibuat. Terdapat dua kinematics yang digunakan dalam animasi karakter yaitu *Forward Kinematics* (FK) dan *Inverse Kinematics* (IK).

Untuk menggerakkan karakter dengan *Forward Kinematic* (FK), anda memutar setiap sendi yang ada pada karakter sampai mendapatkan posisi yang diinginkan. [3]


Gambar 4. Foward Kinematic (FK)

Dengan menggunakan *Inverse Kinematic* (IK) anda dapat membuat struktur kontrol ekstra untuk jaringan sendi tertentu seperti tangan dan kaki. Dengan IK memungkinkan anda untuk menganimasikan atau menggerakkan seluruh sendi dengan memindahkan manipulator tunggal. [3]

Gambar 5. Inverse Kinematics (IK)

### 1.5.4 Constraints

*Constraints* adalah sebuah cara pengaturan otomatis pada posisi sebuah obyek, skala, atau orientasi. Sebuah *constraint* dibentuk dari hubungan langsung antara sumber dan atribut *translate* atau *rotate* obyek target. [4]

[2] Menurut Dariush Derakhshani dalam bukunya yang berjudul Introducting Maya 8, jenis-jenis *Constraint* antara lain :

1. Point Constraint
2. Orient Constraint
3. Scale Constraint
4. Parent, Tangent, and Pole Vector Constraint
5. Aim Constraint
6. Geometry and Normal Constraint

## 2. Pembahasan

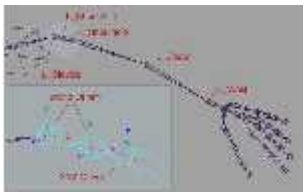
### 2.1 Mendefinisikan Kebutuhan

Kebutuhan akan sebuah alat (*tool*) untuk proses rigging karakter dalam pembuatan film animasi 3D sangat

dibutuhkan. Mengingat jika melakukan rigging secara manual membutuhkan waktu yang cukup lama.

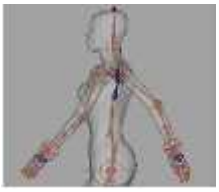
### 2.2 Rancangan Alat Rigging Otomatis

Dalam hal ini akan dibahas mengenai rigging untuk tangan (*hand rig*). Jika tidak menggunakan alat rigging otomatis, maka harus dibuat manual *step by step*. Hal yang pertama kali dilakukan membuat *joints* untuk tangan karakter.



Gambar 6. Joints for Hand

Kemudian baru menentukan apakah akan dianimasikan dengan menggunakan *Foward Kinematic* (FK) atau *Inverse Kinematic* (IK). Misalnya ingin menganimasikan menggunakan IK, maka harus memilih terlebih dahulu *IK Handle Tool*. Kemudian klik pada "*L_Clavial*" *joint* kemudian klik "*L_Wrist*" kemudian baru dapat dianimasikan. Seperti contoh gambar di bawah ini :



Gambar 7. Animasi Tangan

Hal tersebut harus dilakukan berulang kali setiap pembuatan karakter. Oleh karena itu dirancanglah sebuah alat untuk melakukan rigging karakter secara otomatis. Berikut ini adalah *source code* untuk proses rigging tangan.

```
def copyA(each):
 cmds.group(each)
 selParent = cmds.listRelatives(each,
 parent=True, f=True)
  cmds.rename(selParent,'attach_'+each)
 cmds.select(clear=True)
 parent=cmds.listRelatives(each, parent =
 True, f = True)
 attc.append(parent[0])
 cmds.select(each, add = True)
 cmds.select(parent[0], add = True)
 cmds.copyAttr(values = True)

 cmds.xform(each, t=(0, 0, 0),
           ro=(0, 0, 0), s=(1, 1, 1))
 cmds.makeIdentity(parent,apply=True,
                   scale = True)
```

Fungsi di atas difungsikan untuk menduplikasi atribut *translate*, *rotate* dan *scale* masing-masing obyek. Fungsi tersebut juga memberikan nilai default 0 baik atribut *translate* dan *rotate* serta nilai 1 untuk atribut *scale*.

```
def PLACE_LOC(selobj, rad):
 ab = cmds.circle(c=[0, 0, 0],
 nr=[1, 0, 0],sw=360, r=rad,
 d=3, ut=0, tol=0.01, s=8, ch=0)
 CIRCLE.append(ab[0])
 a =  cmds.xform(selobj, t=True,
      ws=True, q=True)
 b =  cmds.xform(selobj, ro=True,
      ws=True, q=True)
 cmds.xform(t=a)
 cmds.xform(ro=b)
 cmds.select(cl=True)
```

Fungsi tersebut digunakan untuk membuat *controller joints* pada bagian tangan karakter.

```
def mag(v):
 m = math.sqrt((v[0] * v[0]) + (v[1] *
     v[1]) + (v[2] * v[2]))
 return m

def norm(x, y, z):
 v = []
 m = mag((x, y, z))
 v.append(x / m)
 v.append(y / m)
 v.append(z / m)
 return v

def POLEV():
 sel=cmds.ls(sl=True,type='ikHandle',
             l=True)
 for ikHandleName in sel:
 px = cmds.getAttr('%s.poleVectorX' %
                   ikHandleName)
 py = cmds.getAttr('%s.poleVectorY' %
                   ikHandleName)
 pz = cmds.getAttr('%s.poleVectorZ' %
                   ikHandleName)
 nv = norm(px, py, pz)
 tempNode = cmds.connectionInfo(
 '%s.endEffector'%ikHandleName,
 sourceFromDestination=True)

 buffer = tempNode.split('.')
 #numTokens = len(buffer)

 endEffectorName = buffer[0]
 parentJoint = cmds.pickWalk(
               endEffectorName,
               d='up')
 parentJointPos = cmds.xform(
                  parentJoint, q=True,
                  ws=True)
 ikHandlePos = cmds.xform(
 ikHandleName,q=True,t=True,ws=True)
 pl = (mag(ikHandlePos) +
       mag( parentJointPos)) / 3
 locName=cmds.spaceLocator(p=(nv[0]*pl
         +parentJointPos[0],nv[1]*pl
         +parentJointPos[1],nv[2]*pl
         +parentJointPos[2]))
```

```
PLOC.append(locName[0])

#cmds.select(locName, replace=True)
cmds.CenterPivot(locName)
cmds.poleVectorConstraint(locName,
ikHandleName)
```

Kedua fungsi tersebut difungsikan untuk menganimasikan karakter 3D menggunakan *pole vector constraint*. Dengan menggunakan *pole vector constraint* maka tingkat kesalahan *joints* dapat diperkecil.

```
PLOC = []
CIRCLE = []
sj = cmds.ls(sl=True, l=True)
dup = cmds.duplicate(sj[0], rc=True,
n='ik_joint')
lan = len(dup)
cmds.select(dup[0])
cmds.select(dup[lan - 2], add=True)
ik = cmds.ikHandle(sol='ikRPsolver')

POLEV()
PLACE_LOC(dup[lan - 2], 2)
copyA(CIRCLE[0])
cmds.parent(ik[0], CIRCLE[0])
cmds.select(dup[-2])
cmds.select(dup[-1], add=True)
ik02=cmds.ikHandle(sol='ikRPsolver')
cmds.select(ik02[0])
cmds.parent(ik02[0], CIRCLE[0])
print 'ikHandle Done'
```

baris kode diatas difungsikan untuk membuat pergerakan tangan dengan *Inverse Kinematic* (IK).

```
nodeLS=['multiplyDivide','condition',
       'setRange','reverse']
SHLS = []
for obj in nodeLS:
sh = cmds.shadingNode(obj,
     asUtility = True)
SHLS.append(sh)
cmds.setAttr(SHLS[1]+'.operation',3)
cmds.connectAttr(SHLS[1] + '.outColorR'
,SHLS[0] + '.input1X')
cmds.setAttr(SHLS[0] +'.operation',2)
cmds.connectAttr(SHLS[0] + '.outputX',
SHLS[2] + '.maxX')
cmds.setAttr(SHLS[2] + '.valueX', 1)
m = cmds.listRelatives(dup[lan - 2],
    parent = True, f=True)
s = cmds.xform(dup[0], t=True,
    ws=True, q=True)
e = cmds.xform(dup[lan - 2], t=True,
    ws=True, q=True)
mid = cmds.xform(m, t=True, ws=True,
      q=True)

Dis01=cmds.distanceDimension(sp=s,
                             ep=e)
Dis02=cmds.distanceDimension(sp=mid,
                             ep=s)
Dis03=cmds.distanceDimension(sp=e,
                             ep=mid)
ConLS = cmds.listConnections(Dis01)
ConLS1 = cmds.listConnections(Dis02)

for locA in ConLS:
cmds.setAttr(locA + '.visibility', 0)
cmds.setAttr(ConLS1[0]+'.visibility',
             0)
cmds.pointConstraint(dup[0],ConLS[0])
cmds.parent(ConLS[1], CIRCLE[0])
cmds.pointConstraint(m, ConLS1[0])

cmds.connectAttr(Dis01 + '.distance',
SHLS[1] + '.secondTerm')
cmds.connectAttr(Dis01 + '.distance',
SHLS[1] + '.colorIfFalseR')

ab = cmds.getAttr(Dis02 +'.distance')
bc = cmds.getAttr(Dis03 +'.distance')

cc = ab + bc

cmds.setAttr(SHLS[1]+'.firstTerm',cc)
cmds.setAttr(SHLS[1]+.colorIfTrueR,c)
cmds.setAttr(SHLS[0]+'.input2X',cc)

selectbone(dup[0])
ADDATTR(CIRCLE[0],'IK_stretch','bool'
cmds.connectAttr(CIRCLE[0]+'.IK_stretc',
SHLS[2] + '.oldMaxX')
print 'IK stretch Done'
```

Baris kode tersebut difungsikan agar ketika lengan ditarik, masing-masing joints pada tangan dapat bertambah panjang secara otomatis dengan ukuran yang proposional karena sudah dihitung menggunakan logika matematika tanpa harus diatur satu per satu.

```
dup02 = cmds.duplicate(sj[0], rc=True,
n='fk_joint')
FKctrl = []
grpLS = []
for obj in dup02[0:-1]:
ab = cmds.circle(c=[0, 0, 0], nr=[1, 0,
0], sw=360, r=3,
d=3, ut=0, tol=0.01, s=8, ch=0)
FKctrl.append(ab[0])

a=cmds.xform(obj,t=True,ws=True,
             q=True)
b=cmds.xform(obj,r=True,ws=True,
             q=True)
cmds.xform(ab, t=a)
cmds.xform(ab, ro=b)

for i in FKctrl:
cmds.group(i)
selParent = cmds.listRelatives(i,
            parent = True, f=True)
sm=cmds.rename(selParent,'attach_'+i)
grpLS.append(sm)
cmds.select(clear=True)
parent = cmds.listRelatives(i,
         parent = True, f = True)
cmds.select(i, add=True)
cmds.select(parent[0], add=True)
cmds.copyAttr(values=True)
cmds.xform(i, t=(0, 0, 0),
```

```
          ro=(0, 0, 0), s=(1, 1, 1))
cmds.makeIdentity(parent, apply=True,
                  scale=True)
```

Baris kode di atas difungsikan untuk membuat *Foward Kinematics* (FK) pada lengan karakter.

```
cmds.setAttr(locA[0]+'.visibility',0)
cmds.setAttr(dup[0]+'.visibility',0)
cmds.setAttr(dup02[0]+'.visibility,0)
cmds.setAttr(Dis01+'.visibility',0)
cmds.setAttr(Dis02+'.visibility',0)
cmds.setAttr(Dis03+'.visibility', 0)
cmds.setAttr(ik[0]+'.visibility', 0)
cmds.setAttr(ik02[0]+'.visibility',0)
attrLS = ['.tx', '.ty', '.tz',
          '.rx', '.ry', '.rz',
          '.sx', '.sy', '.sz', '.v']

for i in attrLS[6:10]:
cmds.setAttr(CIRCLE[0] + i,lock=True,
keyable=False, channelBox=False)

for i in attrLS:
cmds.setAttr(CIRCLE[1]+i,lock=True,
keyable=False, channelBox=False)

for i in attrLS:
cmds.setAttr(locA[0] + i,lock=True,
keyable=False, channelBox=False)

for i in attrLS[3:]:
cmds.setAttr(PLOC[0] + i, lock=True,
keyable=False, channelBox=False)

for i in attrLS[6:]:
for M in FKctrl:
cmds.setAttr(M + i, lock = True, keyable
= False, channelBox=False)

for i in attrLS[:3]:
for M in FKctrl:
cmds.setAttr(M + i,lock=True, keyable
=False,channelBox=False)

cmds.connectAttr(CIRCLE[1] + '.IKtoFK',
SHLS[3] + '.inputX')
cmds.connectAttr(SHLS[3] + '.outputX',
attc[0] + '.visibility')
cmds.connectAttr(CIRCLE[1] + '.IKtoFK',
grpLS[0] + '.visibility')
```

Setelah semua proses dilakukan, langkah yang selanjutnya adalah mengunci dan menyembunyikan atribut-atribut yang telah dibuat. Hal tersebut difungsikan untuk menjaga konsistensi hirarki dari karakter 3D.

### 2.3 Cara Kerja Alat Rigging Otomatis

Berikut ini adalah hasil alat rigging otomatis yang telah dirancang.


Gambar 8. Alat Rigging Otomatis

Pada alat rigging otomatis yang dirancang, animator tidak perlu membuat kerangka dari karakter yang dibuat, karena telah tersedia alat "Import Template".


Gambar 9. Tool for Import Skeleton

Dengan menggunakan alat tersebut, maka secara otomatis akan dibuatkan kerangka dari karakter yang akan dibuat.

Gambar 10. Kerangka Karakter

Dengan menggunakan alat "*Left-Right*" atau "*Right-Left*" maka memudahkan animator untuk memposisikan anggota badan sebelah kanan sama dengan posisi anggota badan sebelah kiri, begitu pula sebalikanya.


Gambar 11. *Tool Left-Right and Tool Right-Left*

Sebagai contoh, animator telah mengatur posisi tangan sebelah kanan dari karakter, maka dengan menggunakan alat "*Right-Left*" maka posisi tangan sebelah kiri juga akan berubah menyesuaikan dengan posisi tangan sebelah kanan.

Gambar 12. Memposisikan tangan kiri seperti posisi tangan kanan

Pada bagian wajah karakter dirancang alat untuk rigging wajah karakter otomatis. Dengan menggunakan alat "*Facial GUI*".


Gambar 13. *Tool for Facial GUI*

Dengan alat tersebut, maka secara otomatis akan dibuatkan template dari wajah karakter yang akan dibuat.

Gambar 14. *Facial GUI*

Dengan menggunakan alat tersebut animator tidak perlu membuat animasi gerakan dari karakter yang dibuat :

Gambar 15. Hasil Akhir

Gambar 16. Rigging Karakter (depan)

Gambar 17. Rigging Karakter (belakang)

## 3. Kesimpulan dan Saran

Alat rigging yang dikembangkan merupakan hasil modifikasi dari alat yang telah disediakan oleh *software* Autodesk Maya. Dengan adanya alat tersebut proses rigging karakter dapat dilakukan lebih cepat dengan meminimalisir kesalahan-kesalahan yang sering dilakukan pada saat rigging karakter seperti hirarki yang tidak konsisten dan lain sebagainya.

Diperlukan pengembangan lebih lanjut mengenai alat rigging yang telah dirancang sehingga dapat memberikan manfaat secara maksimal dan lebih memudahkan pekerjaan animator.

## Daftar Pustaka

[1] Maraffi, C., *Maya Character Creation Modeling and Anination Controls*, Indianapolis: New Riders Publishing, 2004.
[2] Derakhshani, D., *Introducing Maya 8 : 3D for Beginners*, Indianapolis: Wiley Publishing, Inc., 2006.
[3] *Lesson 1 : Skeletons and Kinematics. (2013,* November 2). Diambil kembali dari Autodesk Maya Documents 2010: http://download.autodesk.com/us/maya/2010help/index.html?url=Skeletons_and_kinematics_Creating_joints.htm,topicNumber=d0e20588.
[4] Maestri, G., & Larkins, M., "MAYA 8 AT A GLANCE", Indianapolis: Wiley Publishing, Inc., 2006.

## Biodata Penulis

**Aryanto Yuniawan,** memperoleh gelar Sarjana Komputer (S.Kom), Program Studi Teknik Informatika STMIK AMIKOM Yogyakarta pada tahun 2010. Saat ini masih menempuh program Pasca Sarjana pada Program Studi Magister Teknik Informatika STMIK AMIKOM Yogyakarta dan sebagai Managing Directur PT Mataram Surya Visi (MSV).

**Roy,** memperoleh gelar Ahli Madya Komputer (A.Md), Program Studi Desain/Komunikasi Visual Universitas Kristen Petra Surabaya pada tahun 2006. Saat ini menjadi Technical Director PT Mataram Surya Visi (MSV).

**Stevi Ema Wijayanti**, memperoleh gelar Sarjana Komputer (S.Kom) Program Studi Sistem Informasi STMIK AMIKOM Yogyakarta pada tahun 2013. Saat ini sedang menempuh Program Pasca Sarjana Program Studi Magister Teknik Informatika STMIK AMIKOM Yogyakarta dan sebagai tim Research & Development (R&D) PT Mataram Surya Visi (MSV) sekaligus Dosen Pengajar di STMIK AMIKOM Yogyakarta.