swap constraint akan mengikat objek dengan *locator* yang diciptakan sebagai referensi poin. *Reference locator* memiliki *group offset* yang terikat oleh *base joint* yang akan mendikte posisi dan rotasi pada objek utama.

Fungsi dari *locator* ini juga digunakan untuk memberikan *offset* pada *main object* dengan *joint* dan memberikan fleksibilitas ketika penulis ingin menentukan posisi tertentu pada saat menggunakan *space swap* ini. Pada *global space swap parent* dapat memiliki jumlah yang tidak terbatas, walaupun sering ditemukan tidak lebih dari tiga *parent*.

3. METODE PENCIPTAAN

Deskripsi Karya

Pada laporan ini, penulis akan membahas dua *production tools*, yakni *Global System tools* dan *Match Transformation tools. Global System tools* berfungsi untuk menambah fitur *global transformation* pada *rig*, sedangkan *match transformation* sebagai *tools* pelengkap untuk menghindari *snapping* saat *space swap*. Pengembangan *production tools* ini menggunakan bahasa *programming* Python versi 3.9.6 dengan Maya *commands* dan *framework* QT menggunakan *module* PySide2.

Penciptaan *production tools* ini ditujukan untuk digunakan hanya pada aplikasi Autodesk Maya versi 2017 dan di atasnya, selain itu *tools* ini juga masih belum stabil jika digunakan berdampingan dengan AdvanceSkeleton5. Uji coba *tools* juga diterapkan dalam lingkungan internal Mosmoss studio dan hanya pada divisi *rigging*.

Konsep Karya

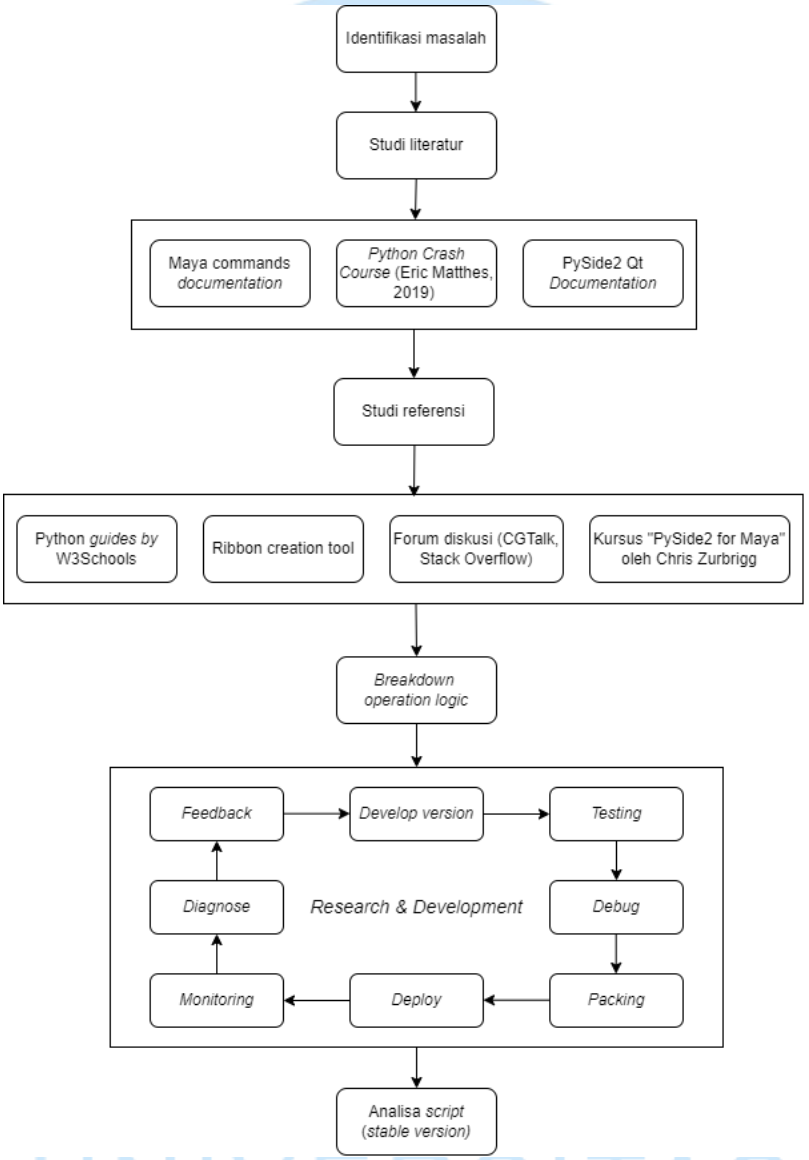
Global system tools dan *Match transformation tools* merupakan salah satu *tools* yang berguna untuk mempersingkat penambahan sistem global pada *rigging*. *Tools* ini diciptakan untuk meminimalisir *human error* yang bisa terjadi, mengingat cara tradisional pada sistem global cukup kompleks karena memiliki tiga sistem yang bisa jadi bertentangan jika prosedur penambahannya salah.

Harapannya *user* bisa menambahkan fitur global dengan berbagai macam cara menambahkan tanpa bersifat destruktif hanya dengan mengisi data yang dibutuhkan dan menjalankan *tools* hanya dengan satu klik.

Match transformation tools diciptakan untuk mendampingi *Global system tools* pada proses *switching* antar *global space swap*. Pada prakteknya, pergantian *Global space swap* akan mengubah posisi semula objek, oleh karena itu diperlukan *Match transformation tools* untuk mengembalikan posisi objek agar tidak terjadi *snapping* saat *switch global space swap*.

Match transformation tools didesain sesederhana mungkin untuk mempermudah *user* dalam menggunakannya. Dengan menggunakan QT *Framework*, penulis dapat menambahkan *icon* gambar dan juga *dynamic widget* pada *tools* ini yang mempermudah indikasi cara pemakaian. Selain itu organisir *layout* dalam *scripting* menjadi lebih *clear* dan juga mudah. Penggunaan Python dengan *module* QT dan *maya commands* merupakan bagian fundamental dalam menciptakan *script* ini.

Tahapan Kerja



Gambar 1. Bagan skematika penelitian pengembangan *production tools*

(Sumber: Dokumentasi Pribadi)

1. Identifikasi masalah

   Berangkat dari kendala penulis karena sering mendapatkan *request* dari *animator* untuk menambahkan fitur *global space swap, global rotation dan translation*. Cara manual bisa dilakukan oleh *rigger* namun memakan waktu yang panjang dan bisa terjadi *human error*. Proses ini biasa memakan waktu 7 menit atau lebih tergantung *parent* (belum termasuk

global rotation dan *translate*). Durasi 7 menit mungkin tidak terlihat besar, tapi jika dilipatkan dengan berapa kali penulis harus melakukan langkah yang sama pada setiap *rig* baru, tentunya akan menjadi sesuatu yang repetitif dan membuang waktu penulis.

Tabel 1. Waktu dan jumlah klik prosedur manual *global system*

	tanpa plugin
Waktu mengerjakan	
Jumlah parent 2	7 menit 24 detik
Jumlah parent 3	8 menit 53 detik
Jumlah parent 4	9 menit 13 detik
Jumlah Klik mouse	
Jumlah parent 2	215 klik
Jumlah parent 3	274 klik
Jumlah parent 4	311 klik
Penamaan	Manual

Penulis juga sulit dalam menemukan *global system tools* yang sesuai dengan kebutuhan, salah satu *tools* yang mirip namun belum memenuhi kebutuhan penulis adalah *Space switch tool* karya Riham toulan. *Tools* ini memiliki UI yang lengkap tetapi masih memiliki beberapa kekurangan bagi penulis seperti, penggunaan *swapping* yang berupa *enum attribute* (berpotensi *snapping* saat *swapping*) dan juga tidak ada fitur *global rotate* dan *global translate* yang terintegrasi dengan *space swap.*

2. Studi Literatur

Sebagai fondasi dasar dalam penulisan *coding* yang benar, penulis membaca *e-book Python Crash Course* karya Eric Matthes (2019). Dari situ penulis memperkuat pengetahuan *python* yang sebelumnya penulis ketahui untuk menulis *code* yang sistematis, mudah dibaca dan sesuai dengan kaidah yang berlaku.

Sebelum mengerjakan program, penulis akan menjabarkan proses penambahan sistem global dan kemudian menggunakan *library maya commands* dan *library* Qt untuk PySide2. Pada umumnya proses pencarian *commands* akan dilakukan bersamaan dengan pengerjaan *script*, sehingga

penulis akan mencari *commands* yang dibutuhkan pada dokumentasi Autodesk Maya di *website* help.autodesk.com.

3. Studi Referensi

Selain menggunakan buku sebagai sumber belajar, penulis mendalami pengetahuan Python melalui *website* W3schools dan Real Python. Sedangkan untuk implementasi Qt *framework,* penulis belajar dari kursus "PySide2 for Maya" yang diajar oleh Chris Zurbrigg.

Ketika mengalami masalah, penulis mencari referensi dari forum diskusi dimana pada forum tersebut juga terdapat permasalahan yang sama dan sudah memiliki solusi untuk masalah tersebut. Biasa penulis menggunakan *website* CG Talk dan Stack Overflow.

Penulis juga terkadang membaca kembali *script* lama yang pernah dibuat penulis. Kasus yang paling sering ditemukan biasanya ada pada cara bagaimana meng-*query* data yang ada pada UI ataupun struktur penulisan dan pengaturan UI Maya.

4. Breakdown *operation logic*

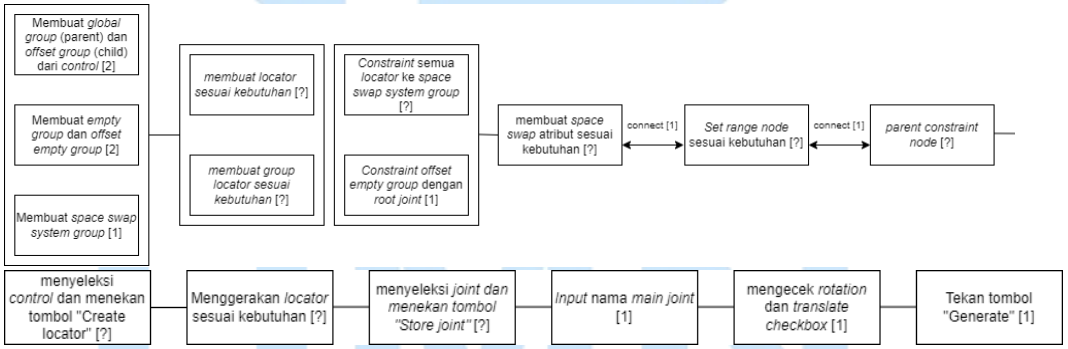
Sebelum mengerjakan *tools* penulis akan menjabarkan struktur *scripting* dan juga variabel-variabel yang dibutuhkan dari *user* maupun *variable runtime* dalam bentuk *pseudocode.* Setelah mengerjakan *pseudocode,* penulis akan mencoba menerjemahkannya ke dalam bahasa Python seperti yang digambarkan di bawah (Gambar 2).

Penjabaran ini masih belum final karena dilakukan sebelum mendesain UI dan menambahkan *operation logic* atau *commands* yang dibutuhkan untuk menambahkan sistem global ke *rig*.

```python
# USER DATA:
main_control = []     # User input
joint_name = []       # User input

# RUNTIME CACHE:
stored_locator = []   # global variable
stored_joint = [] # global variable

# OPERATION LOGIC:
def create_locator(*args):
    "create new locator under global_system_group, and store it to runtime cache"     # Function description

def store_locator(*args):
    "when user press button while joint selection is active, store the joint to stored_joint global variable"
    "store the locator transformation to global variable stored_locator"       # Function description

def global_space_swap():
    if len(stored_joint) < 2:
        "ignore global_space_swap function if user only create 1 locator"      # Function description
    else:
        "run the global space swap system"      # Function description

def global_translation():
    "function to create global translation system"     # Function description

def global_rotation():
    "function to create global rotation system"     # Function description
```

Gambar 2. *Breakdown script*

(Sumber: Dokumentasi Pribadi)



Gambar 3. Prosedur manual *global system*

(Sumber: Dokumentasi Pribadi)

Penulis menjabarkan prosedur dari pembuatan global space swap sampai kepada cara menghindari *system clash*. Langkah ini dilakukan dengan praktek langsung pada aplikasi Maya. Penulis kemudian mencatat langkah ini untuk diurutkan dan dipakai sebagai salah satu pedoman dalam menulis *script,* dalam menghindari kode yang tidak terpakai.

5. *Research & Development*

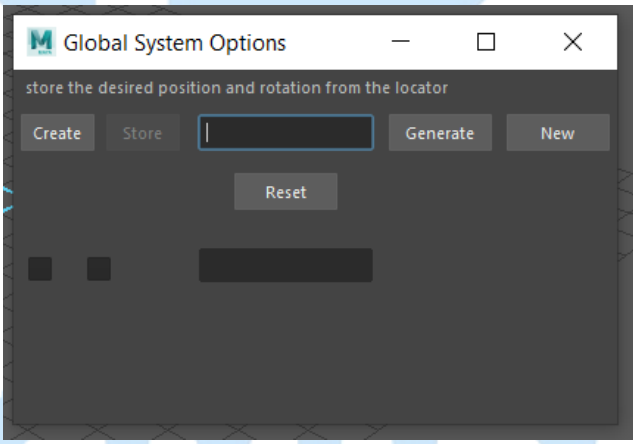
Pada tahap ini penulis akan meng-*developing production tools* dengan *workflow* yang bersifat siklik. Proses pengerjaan *script* ini akan terus

berulang sampai ke versi yang lebih stabil dan siap diimplementasikan pada *workflow* studio (Perlu diingat ada kemungkinan *bug/error* pada versi ini).

Berikut penulis akan menjabarkan proses pengerjaan *production tools* dalam beberapa bagian proses:

a. *Testing*

Setelah pengerjaan *script* yang panjang disertai dengan *runtime debug*, penulis akhirnya akan menyatakan *script* ada pada versi tertentu, biasanya penulis akan menulis versi paling pertama dengan "versi 0.1" dimana *production tools* masih bersifat *beta* dan memiliki banyak kemungkinan *bug* dan *error*.



Gambar 4. *UI Global system tools* versi *beta*

(Sumber: Dokumentasi Pribadi)

Meski demikian penulis biasanya akan mempraktekan *tools beta* ini sesuai dengan fungsinya, dan yang paling sering terjadi akan ada *error* jika penggunaan mulai bervariatif. Pada tahap ini penulis akan terus merevisi *script* sampai dirasa sudah cukup stabil dan akan lanjut ke tahap selanjutnya yaitu, *packing*.

b. *Packing*

Pada tahap ini *source code production tools* akan di-*convert* menjadi Python *file*. Python *file* ini akan diganti nama menjadi "Global system tools" dan "Match transformation tools" untuk mempermudah pencarian *source code* kedepannya. Tidak lupa *file* ini akan diletakkan pada *script folder* Maya.

c. *Deploy*

Setelah proses *import* pada *command line* ataupun *shelf* pada *script* yang sudah di-*packing*, penulis dapat menggunakan *production tools* ini untuk diuji lebih lanjut pada *rigging*. Pada tahap ini *production tools* juga bisa didistribusikan dengan skala kecil mengingat kemungkinan *error* masih cukup besar terutama pada *beta version*.

d. *Monitoring*

Saat *monitoring* penulis akan melakukan observasi pada penggunaan *production tools* dan siap untuk mencatat permasalahan-permasalahan yang kemungkinan muncul.

Pada saat menggunakan *Global System tools* versi *beta*, penulis mencoba memakai fitur *global space swap* untuk menggerakkan *control props*. Hasilnya *rig* ini memiliki fitur *global space swap* tanpa masalah. Tetapi ketika penulis menambahkan *global rotation* ataupun *global translation*, timbul masalah dimana *controller* tidak mengikut *space swap* lagi. Masalah lain juga ditemukan ketika UI *Global system tools* tidak menyimpan *preset* yang sudah ditentukan *user* ketika *user* keluar dari UI.

Penulis juga mendistribusikan *tools* kepada *supervisor* lapangan. Hasilnya, penulis mendapatkan saran untuk mengganti *main joint* menggunakan *global control* untuk *global rotation* dan *translate* karena kesalahan sistem pada *parent.* Perlu diingat semua

masalah yang terjadi merupakan hal wajar pada tahap *monitoring,* dan untuk solusinya akan dilaksanakan pada tahap berikutnya.

e. *Diagnose*

Ketika ditemukan masalah-masalah pada *production tools* yang ada maka pada tahap *diagnose*, penulis akan melihat kembali *script* dan mencoba menentukan penyebab masalah maupun solusi baru

f. *Feedback*

Setelah sumber masalah atau solusi ditemukan, penulis akan mencatatnya dan memulai pengaplikasian *feedback* pada tahap *debug. Feedback* juga bisa didapatkan dari orang lain jika penulis mengerjakan *project* dengan tim.

Berdasarkan hasil *diagnose,* Masalah *global space swap* dengan *global rotation* pada *Global system tools* ada karena *group static* FK *global controller* yang tidak berubah *parent* ketika *space swap* dijalankan. Untuk masalah UI yang tidak mengingat *user preset* ada pada UI *function* yang tidak memiliki *code* yang berfungsi untuk menyimpan preferensi.

g. *Debug*

Proses *debug* dijalankan menggunakan *plug-in* Charcoal editor, seperti pada proses-proses sebelumnya. Setelah mengetahui sumber masalah dari *feedback* yang ada, penulis menjadi lebih paham untuk menggunakan solusi atau *command* yang pada *library.*

Penulis melakukan revisi *script* dengan menambahkan operasi baru dimana *group static* FK akan mengikuti *parent* yang ada pada *global space swap* dengan bantuan metode *for loop* dan *global variable* (Referensi pada gambar 5).

```
# Check how many global space swap from constraint in outliner
temp_locator = cmds.listRelatives("Curve_Locators_Grp", c=1)

for i in range(len(storedParentJoint)):
    cmds.setAttr(driverAttribute,count7)
    cmds.matchTransform(mainFKTextField+"_gSystem_to_gSpace",storedCurveLocators[count8])
    for i in drivenList:
        cmds.setDrivenKeyframe(i,cd=driverAttribute)
    count7+=globalAttrValueRatio
    count8+=1
```

Gambar 5. Salah satu contoh *code debug*

(Sumber: Dokumentasi Pribadi)

Selain itu penulis juga meng-*debug* masalah *preset* yang tidak tersimpan menggunakan *maya command* "cmds.optionVar" pada UI *function*. *Commands* optionVar berguna untuk menyimpan data yang bersifat tetap diantara sesi Maya.

h. Develop version

Setelah *production tools* melewati proses revisi yang panjang, akhirnya *production tools Global space swap* dan *Match transformation* bisa masuk ke versi stabil yang lebih baru atau pada kasus penulis versi 1.0.

Tentunya proses ini bukan yang terakhir karena *workflow* akan berputar lagi. *Production tools* ini akan masuk ke tahap *testing*, *packing* dan *deploy* sehingga *tools* dapat secara berkala didistribusikan ke skala yang lebih besar untuk *monitoring*.

6. Analisa *Script*

Setelah penggunaan dan analisis pada *tools,* penulis mendapat *input* berbagai fitur dan masalah yang bisa dikerjakan untuk versi berikutnya. Beberapa di antaranya (*minor*):

- *Tools* belum bisa bekerja untuk beberapa objek 3D dalam satu kali *setup*

- *Tools* belum secara otomatis menghapus *group* kosong di *outliner* (*Optimize)*