ARC reference modules for representing real characters

Kwan-Hee Yoo

2012. 10. 24

Dept. of Digital Informatics and Convergence
Chungbuk National University

Introduction

- Traditional e-Learning Systems
  - Contents: text, image, video, flash animations
  - Are applied to cooperative learning and self-directed learning techniques
  - Have some problems
    - Most learning is in real world (3D environment)
    - immersion, virtual experience and interaction emerging

Introduction

- 3D Virtual Experience e-Learning System
  - To improve learning immersion, learners and teachers will participate in 3D virtual space.
  - As they are acting in real environment, their actions are appeared in 3D virtual space.

  - One of best examples:

    Sloodle (Second Life Virtual Education Systems + Moodle)
    - Avatars: learning immersion of learners is relative low

3D Virtual Education Services

Participation Methods of Learners

In the 3D Virtual space

Avatars

Avatars and real characters

Real characters

Immersion, virtual experience and interaction emerging

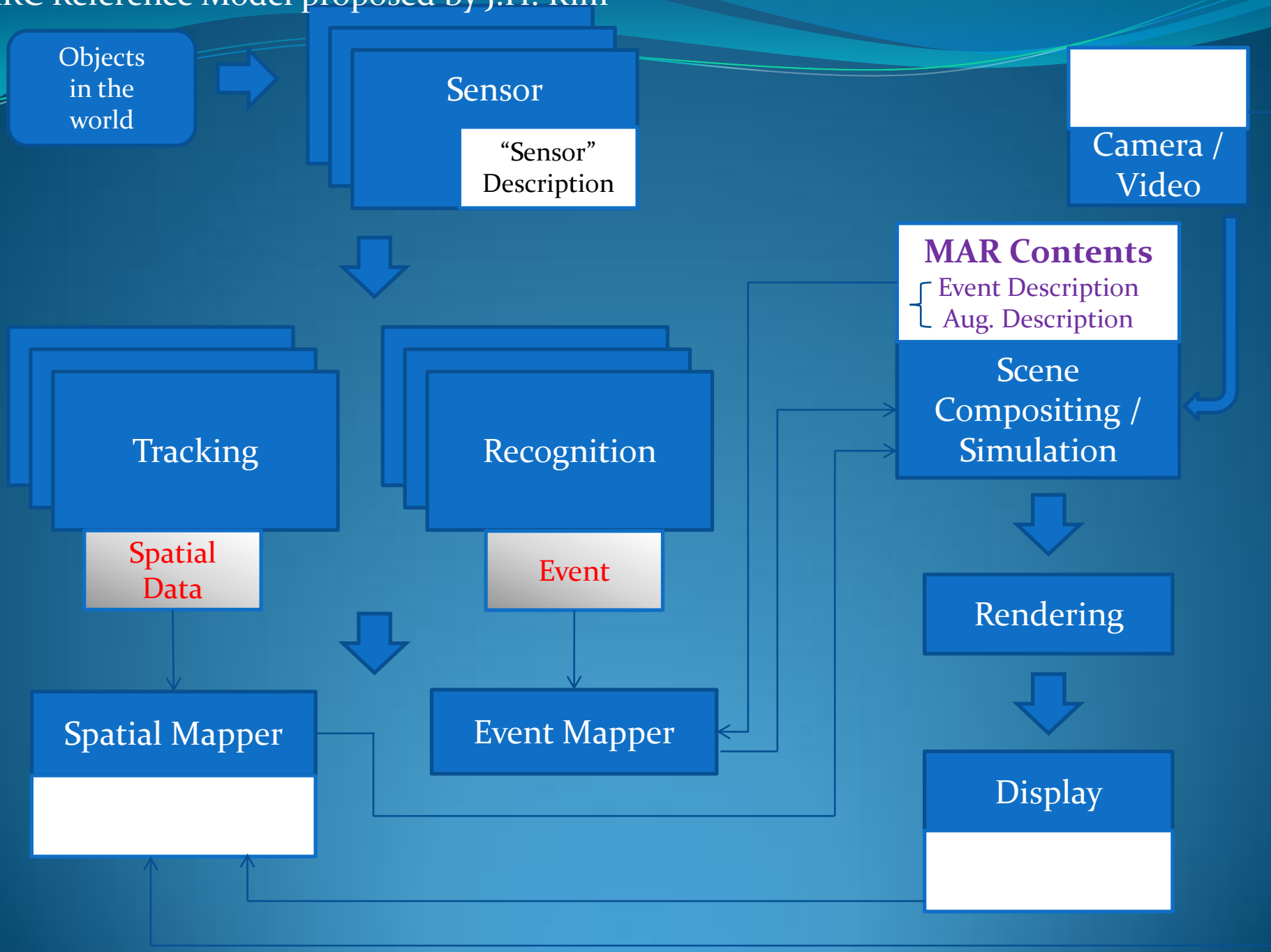
In this paper, we propose

Reference modules for representing real characters in the 3D virtual space.

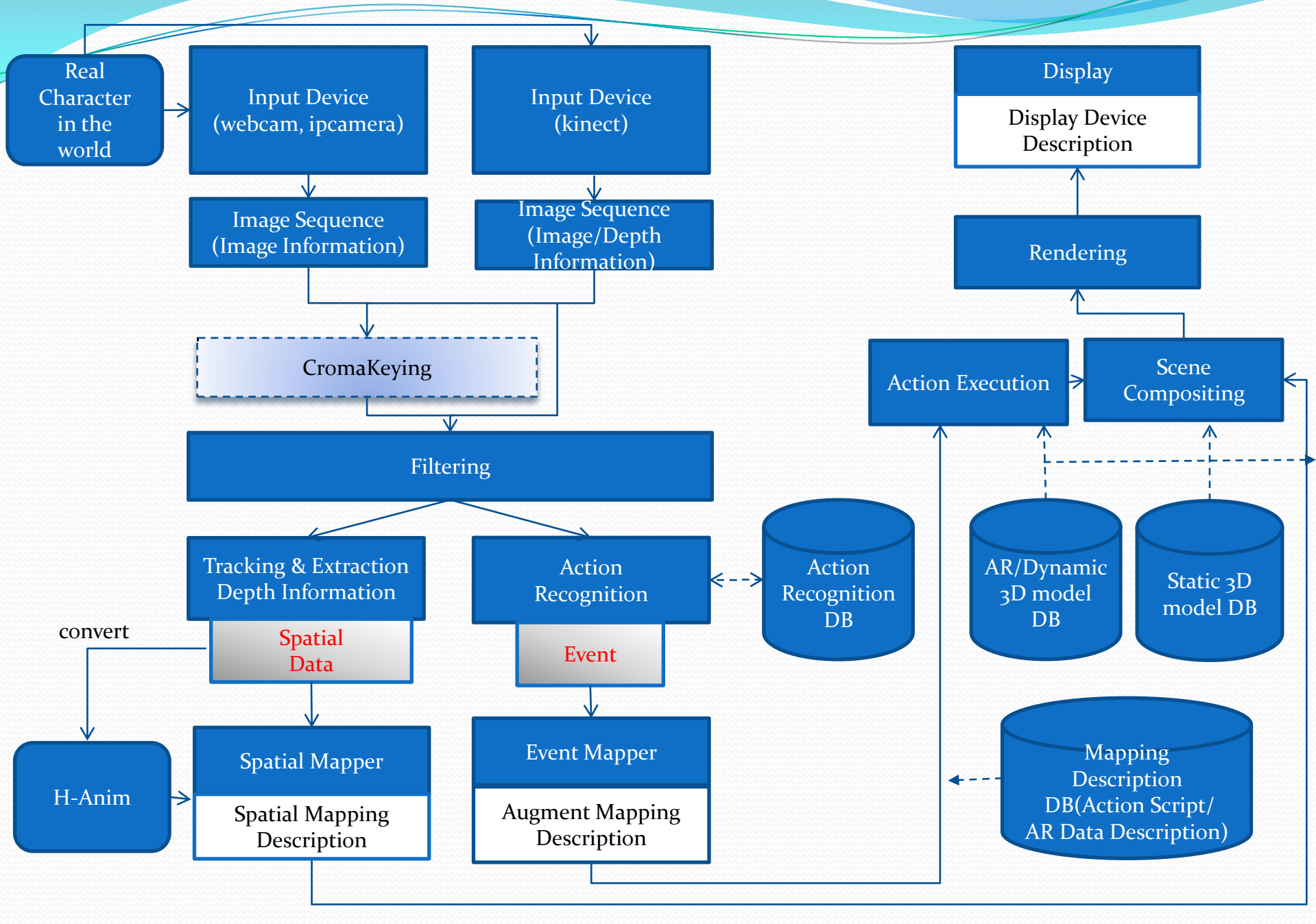
표준추진 현황

- ISO/IEC JTC 1/SC24 WG6 발표(2011년 미국)
- ISO/IEC JTC 1/SC24 WG9 발표(2012년 8월 벨기에 브뤼셀)
  - NP 제안

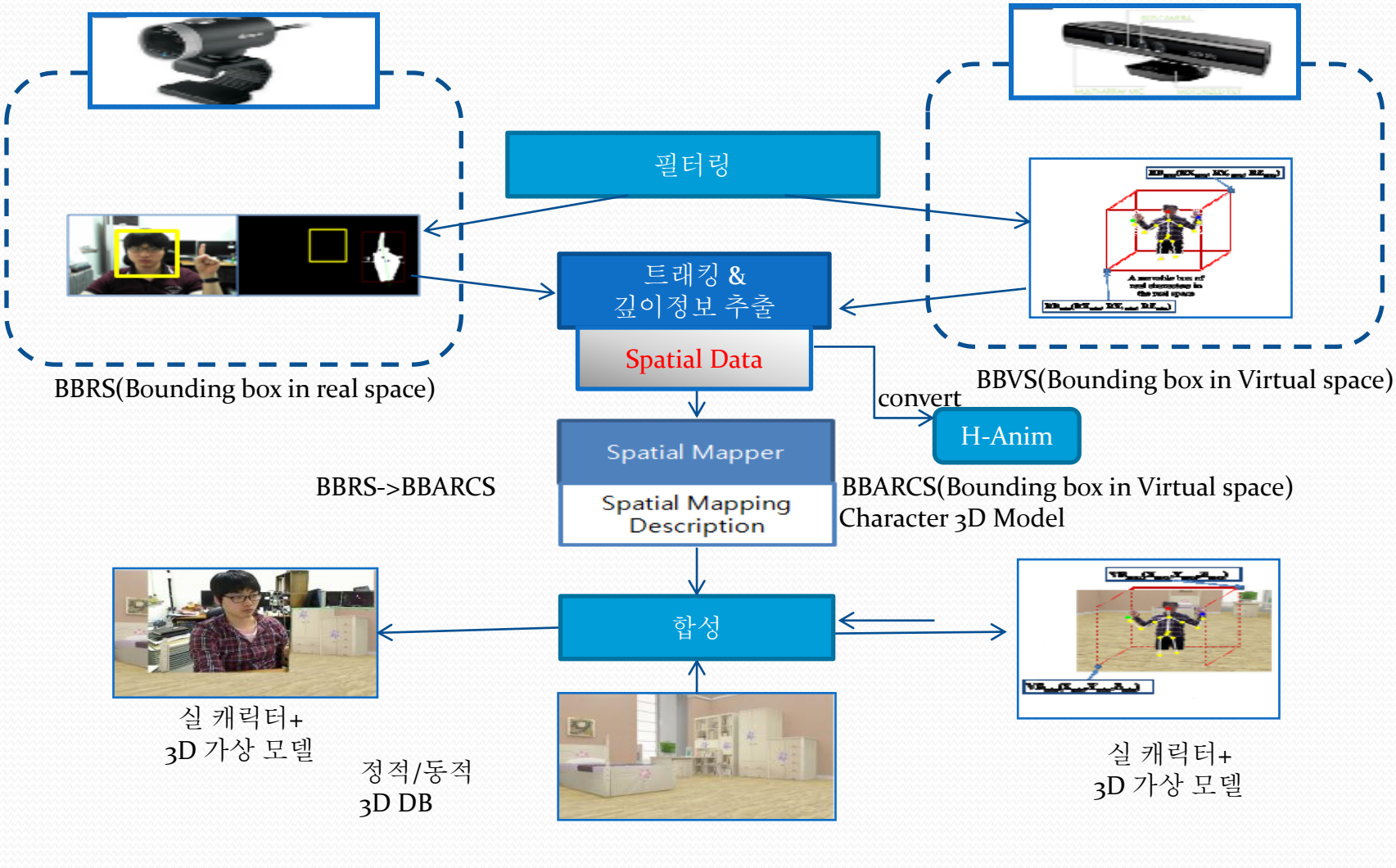
ARC Reference Model proposed by J.H. Kim



ARC Reference Modules for Representing Real Characters proposed by K.H. Yoo



실 캐릭터 표현을 위한 참조 모델

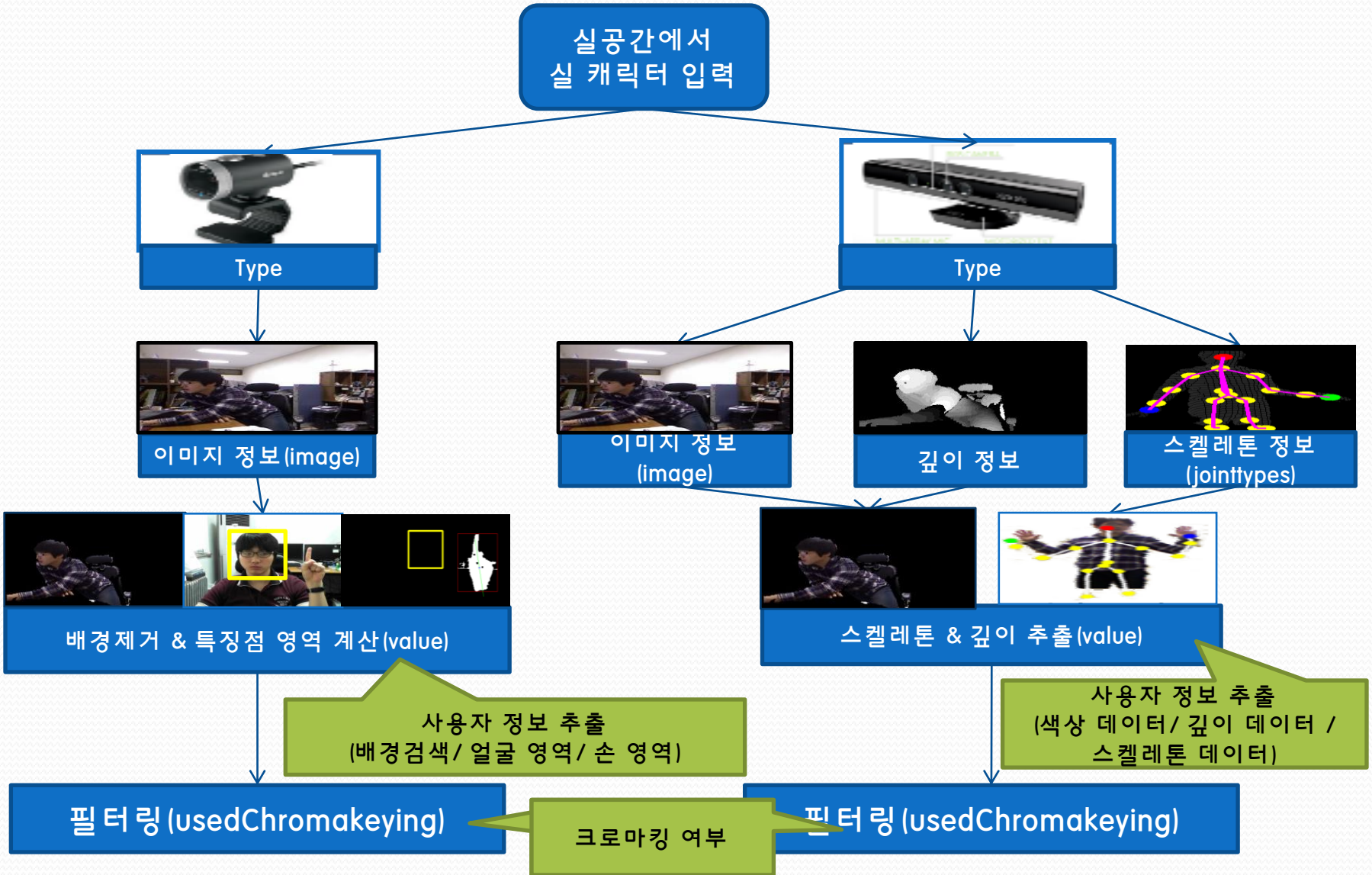


실 캐릭터 입력 센싱 기법

실 공간에서 실 캐릭터를 센싱하기 위한 노드

```
RCSensingDeviceNode
{
  SFString[in]      id                ""
  SFString[in]      description       ""
  SFString[in]      type              camera     // 일반카메라 or 깊이카메라
  SFFloat[in]       fov               45.0
  SFInt [in]        framerate         20         // 초당 프레임 수
  SFImage[out]      image             0 0 0   // 캡쳐 이미지 좌표 값
  MFString[in]      jointypes         ""       // 조인트 포지션의 이름(H-Anim)
  MFVector[out]     values            0 0 0   // 조인트 포지션 깊이 좌표 값
  SFBool[in]        usedChromakeying  false  //크로마 킹 여부(true, false)
}
```

실 캐릭터 입력 센싱 메커니즘



jointtypes(H-Anim)

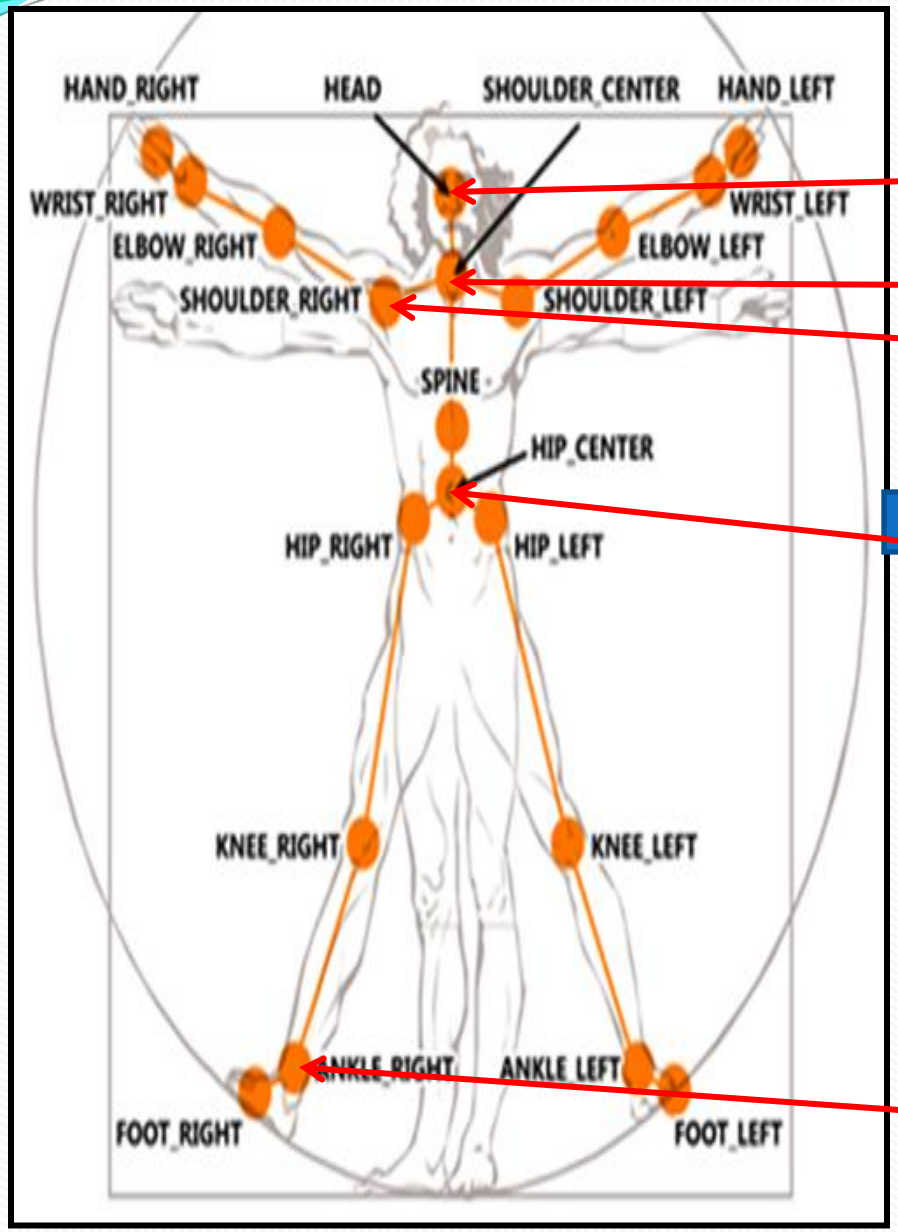
[이명원, 휴먼 모션 데이터 표준화, 2008 웹디지털 콘텐츠 표준화 포럼]

- Humanoid Animation Working Group에서 제정
- H-Amin 200x: ISO/IEC FCD 19774 표준으로 제정(2004)
- 3D 인터넷이 성장하면서 온라인 가상환경에서 인간을 표현하고자 하는 필요성 증대
  - 새로운 휴먼노이드를 손쉽게 제작할 수 있는 저작도구
  - 다양한 방법으로 움직이도록 하기 위한 호환 가능한 휴먼노이드 (humannoid) 라이브러리의 제작이 필요하게 됨
- 인체 구조의 각 관절과 메쉬의 정보를 저장하는 노드들의 집합
  - H-Anim의 구조

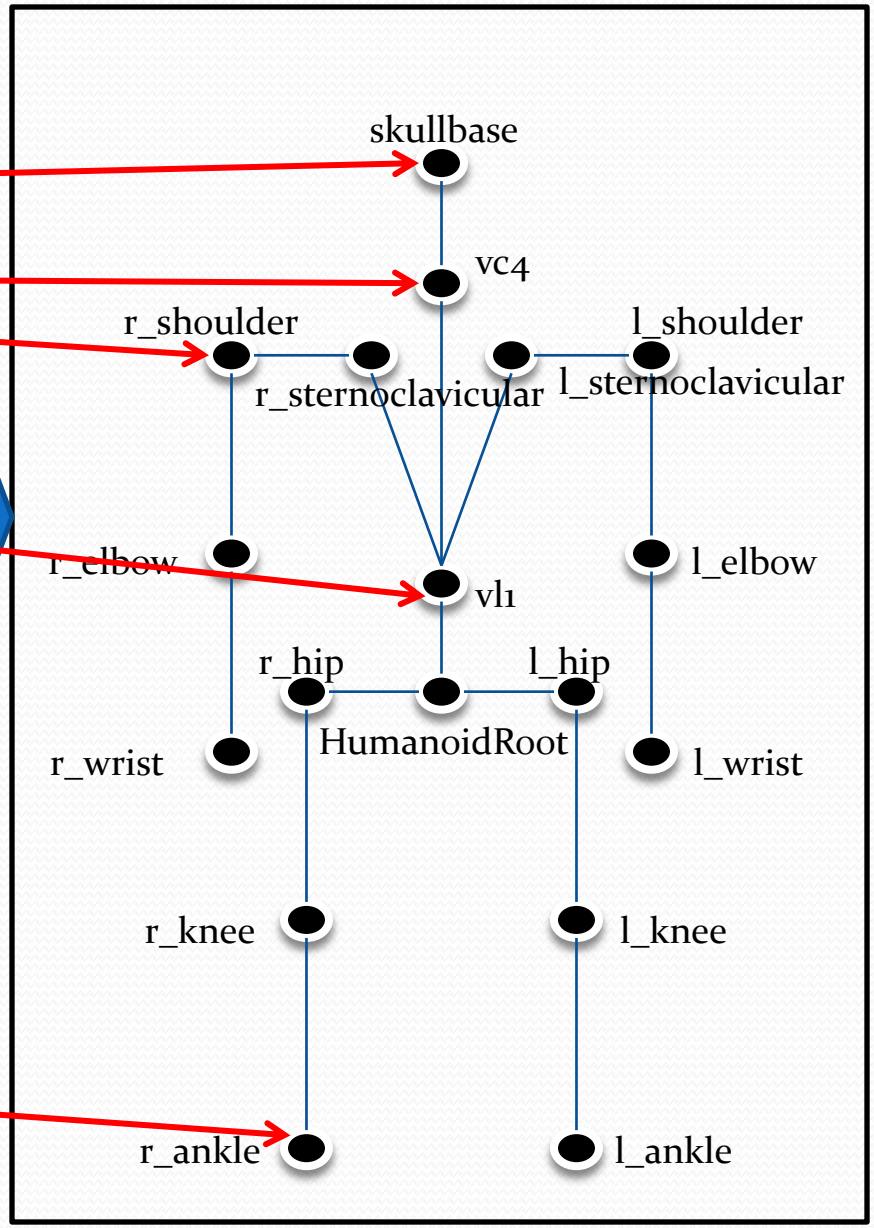
구분	설명
Humanoid	관절, 인체부위, 시야의 참조점을 제공하고 전체 인체모형의 용기 역할을 수행
Joint	인체의 각 관절은 Joint 노드로 표현되며, 이것은 인체의 각 부위와 상위 부위와의 관계를 정의하는데 사용
Segment	모든 신체 부위는 Segment 노드에 저장
Site	의류나 보석 같은 장식들의 부착점, 가상 카메라의 위치를 정의
Displacer	움직임의 범위에 관한 정보 포함

KINECT

H-Anim



Convert



Kinect 구조와 H-Anim 구조 매치

Kinect 관절 이름	H-Anim 관절이름	논문 사용 관절이름
HEAD	skullbase	HEAD
SHOULDER_CENTER	vc4	c_shoulder
SHOULDER_RIGHT	r_shoulder	r_shoulder
SHOULDER_LEFT	l_shoulder	l_shoulder
ELBOW_RIGHT	r_elbow	r_elbow
ELBOW_LEFT	l_elbow	l_elbow
WRIST_RIGHT	r_wrist	r_wrist
WRIST_LEFT	l_wrist	l_wrist
SPINE	vl1	SPINE
HIP_CENTER	HumanoidRoot	c_hip
HIP_RIGHT	r_hip	r_hip
HIP_LEFT	l_hip	l_hip
KNEE_RIGHT	r_knee	r_knee
KNEE_LEFT	l_knee	l_knee
ANKLE_RIGHT	r_ankle	r_ankle
ANKLE_LEFT	l_ankle	l_ankle
-	r_sternoclavicular	r_sternoclavicular
-	l_sternoclavicular	l_sternoclavicular
HAND_RIGHT	-	r_hand
HAND_LEFT	-	l_hand
FOOT_RIGHT	-	r_foot
FOOT_LEFT	-	l_foot

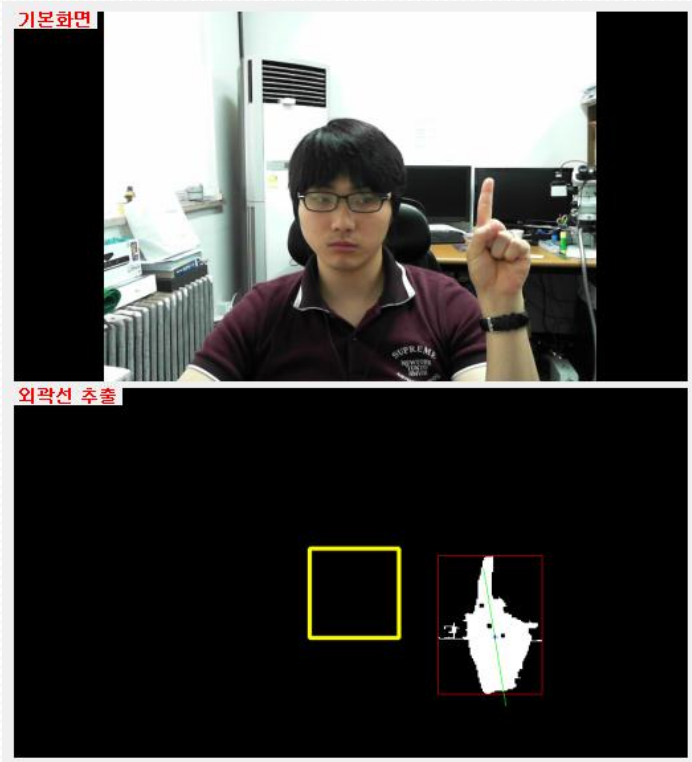
- Chroma Keying
  - 객체와 단색 배경을 분리하여 새로운 영상을 합성시키는 기술
  - 일기예보, 영화 등을 촬영 할 때 많이 사용됨.

실 공간에서 실 캐릭터의 움직이는 영역 설정 방법

실 공간에서 실 캐릭터의 움직이는 영역 설정 노드

```
RCBBRSNode
{
  SFString[in]                id              ""
  SFString[in]                description     ""
  SFVector[in,out]            startpoint      0 0 0
  SFVector[in,out]            endpoint        320 240  0
  RCSensingDeviceNode[in]     sensingDevice     ""
}
```

실 공간에서 실 캐릭터의 움직이는 영역 설정(RCBBRS) - 일반카메라

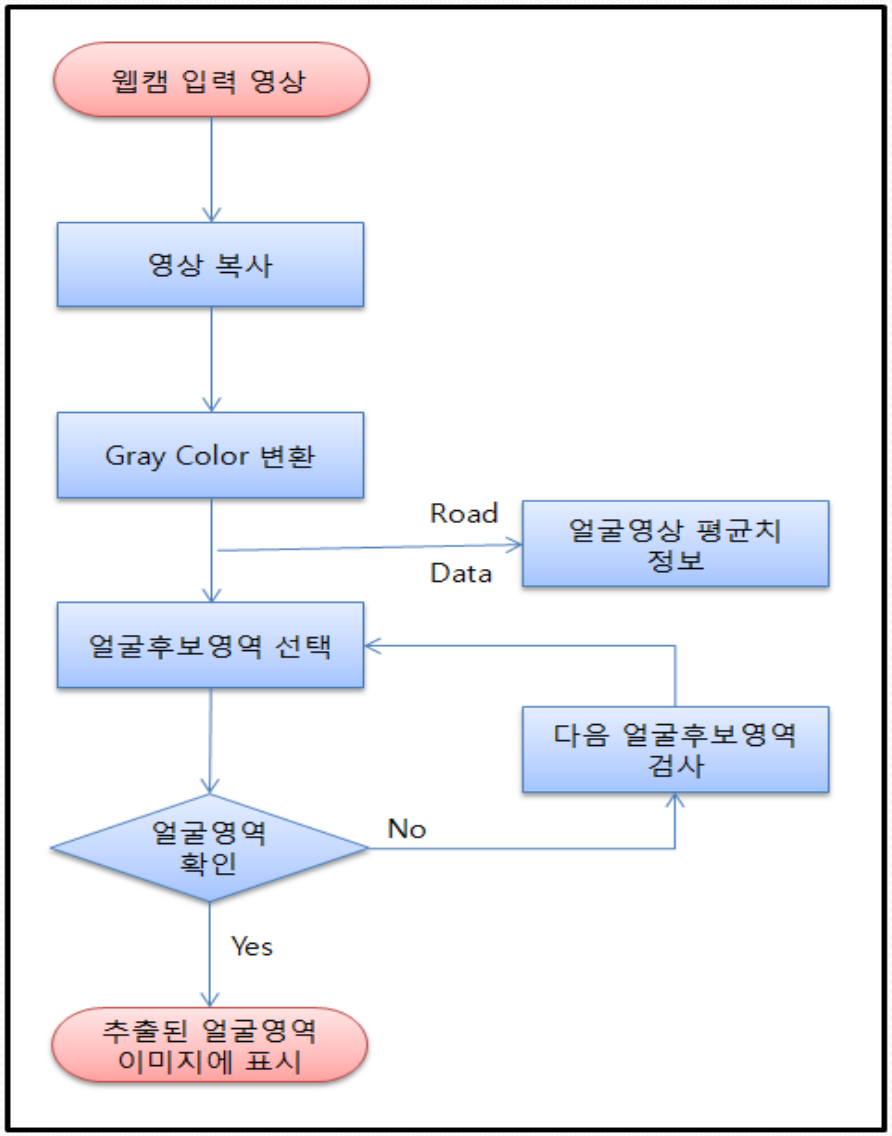


$$FHA_t = w_f x FA_t + w_h x HA_t$$
$$RZ_t = (FHA_t - RZ_{min}) / (RZ_{max} - RZ_{min})$$

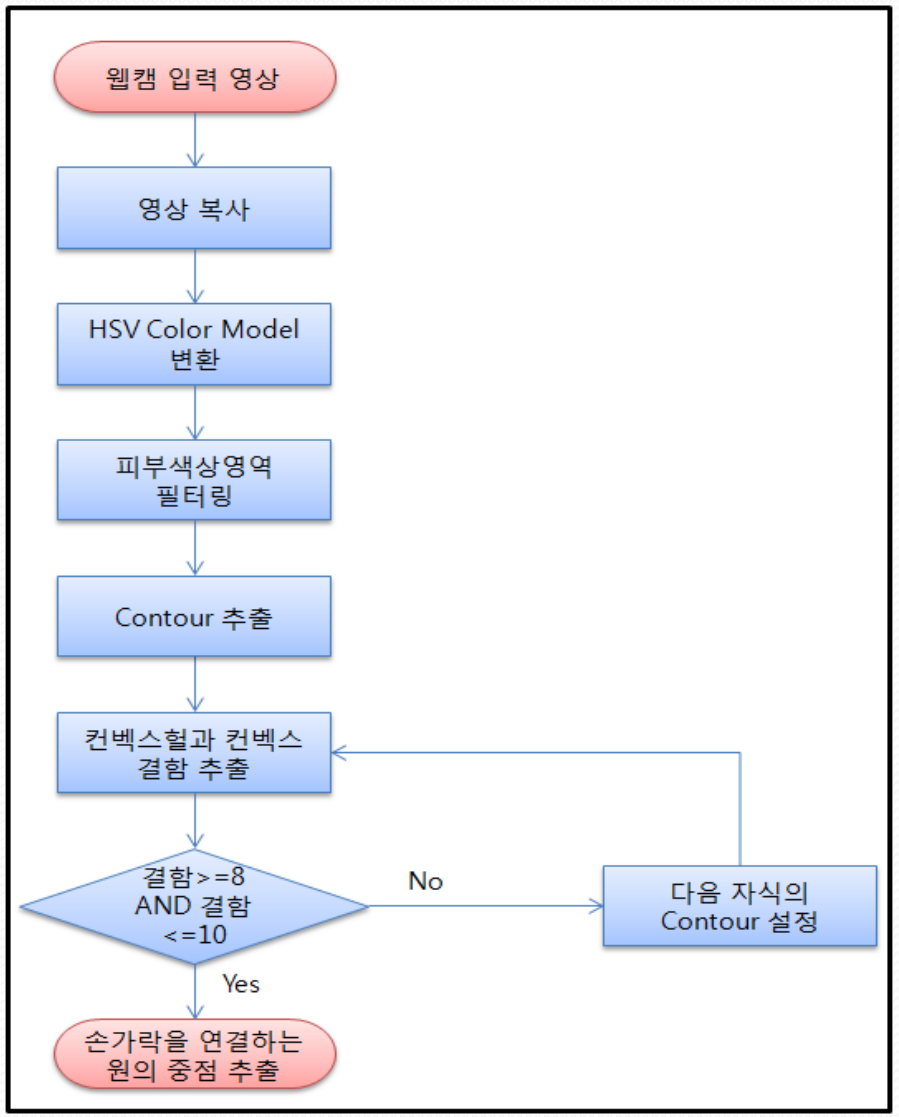
이미지 사이즈 = Iw, Ih
RX_{min} = o, RY_{max} = o, RX_{min} = Iw, RY_{max} = Ih

- FA_t, HA_t : 실 캐릭터의 얼굴과 손영역이 가장 작은 사각형 공간에 있을 때를 표현함
- t : 시간
- w_f, w_h : 얼굴과 손의 가중치
- RZ_{min} : 카메라에서 실 캐릭터가 멀어졌을 때
- RZ_{max} : 카메라에서 실 캐릭터가 가까워졌을 때

얼굴 영역 추출

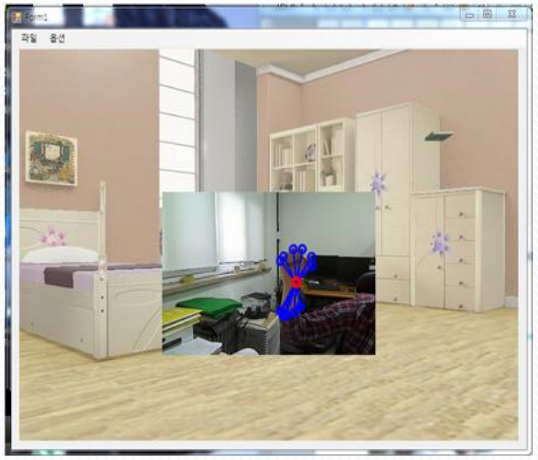


손 영역 추출



최대 얼굴 영역에 대한 정보 추출

최소 얼굴영역에 대한 정보 추출

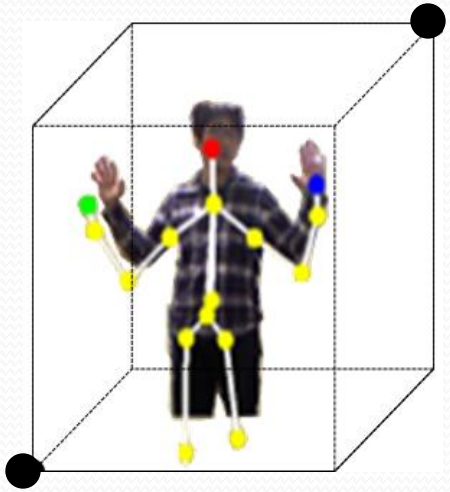


손가락을 접었을 때 손위치 추적 결과

손가락을 폈을 때 손위치 추적 결과

실 공간에서 실 캐릭터의 움직이는 영역 설정(RCBBRS) – 깊이 카메라

$RCBBRS_{max}(RX_{max}, RY_{max}, RZ_{max})$



$RCBBRS_{min}(RX_{min}, RY_{min}, RZ_{min})$

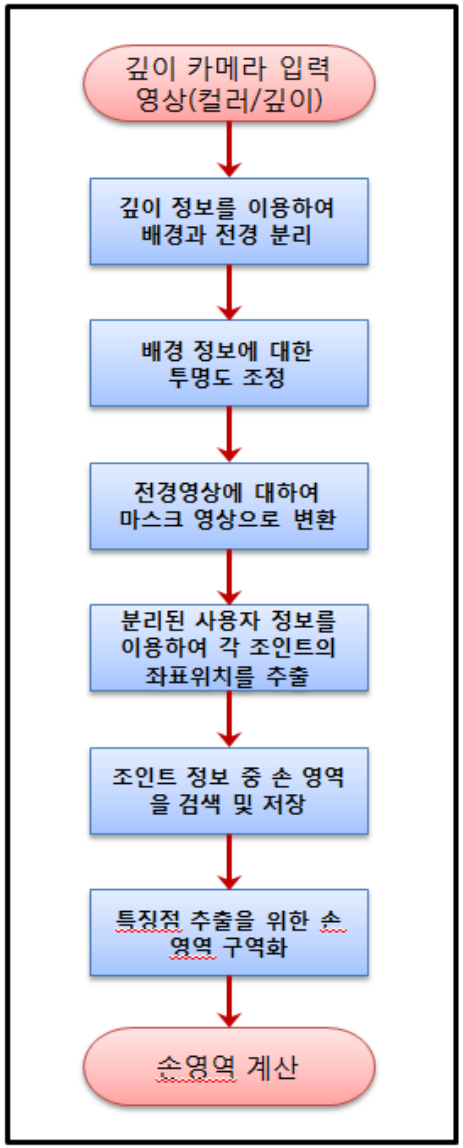
$$HLR_t = (H_t + L_t + R_t)/3.0$$
$$RZ_t = (HLR_t - RZ_{min}) / (RZ_{max} - RZ_{min})$$

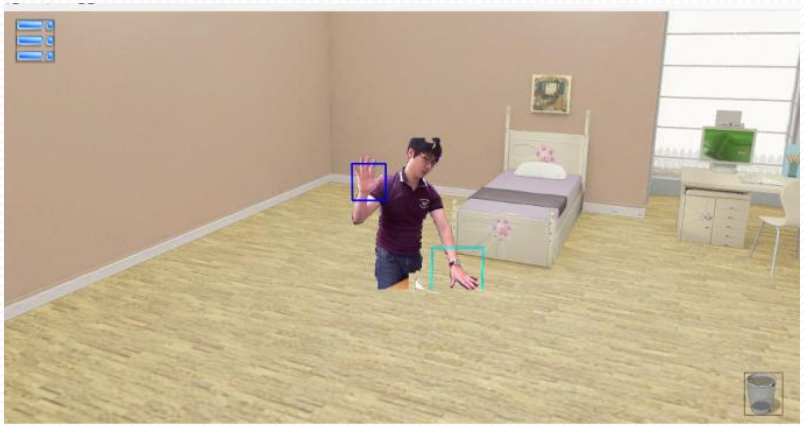
이미지 사이즈 = Iw, Ih

RX_{min} = o, RY_{max} = o, RX_{min} = Iw, RY_{max} = Ih

- RZ_{min} : 카메라에서 실 캐릭터가 멀어졌을 때
- RZ_{max} : 카메라에서 실 캐릭터가 가까워졌을 때
- H_t: 한 개의 손, L_t: 왼 손, R_t: 오른 손

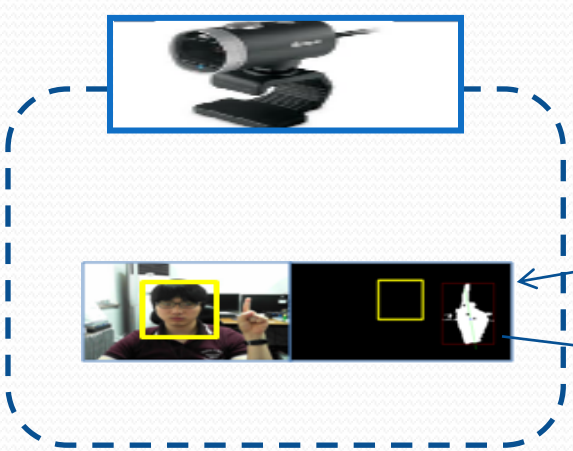
손 영역 추출

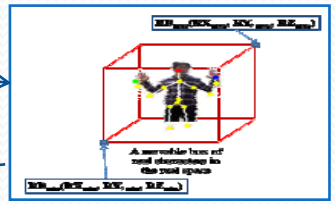

깊이 카메라 입력 영상(컬러/깊이)
깊이 정보를 이용하여 배경과 전경 분리
배경 정보에 대한 투명도 조정
전경영상에 대하여 마스크 영상으로 변환
분리된 사용자 정보를 이용하여 각 조인트의 좌표위치를 추출
조인트 정보 중 손 영역을 검색 및 저장
특징점 추출을 위한 손 영역 구역화
손영역 계산



손 영역 계산

가상공간에서 실 캐릭터의 움직이는 영역 설정





필터링

트래킹 &
깊이정보 추출

Spatial Data

BBRS(Bounding box in real space)

BBVS(Bounding box in Virtual space)

convert

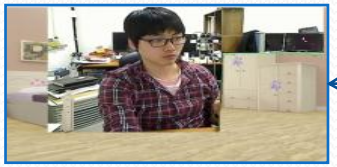
H-Anim

Spatial Mapper

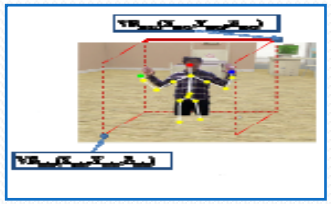
Spatial Mapping
Description

BBRS->BBARCS

BBARCS(Bounding box in Virtual space)
Character 3D Model



합성



실 캐릭터+
3D 가상 모델

정적/동적
3D DB

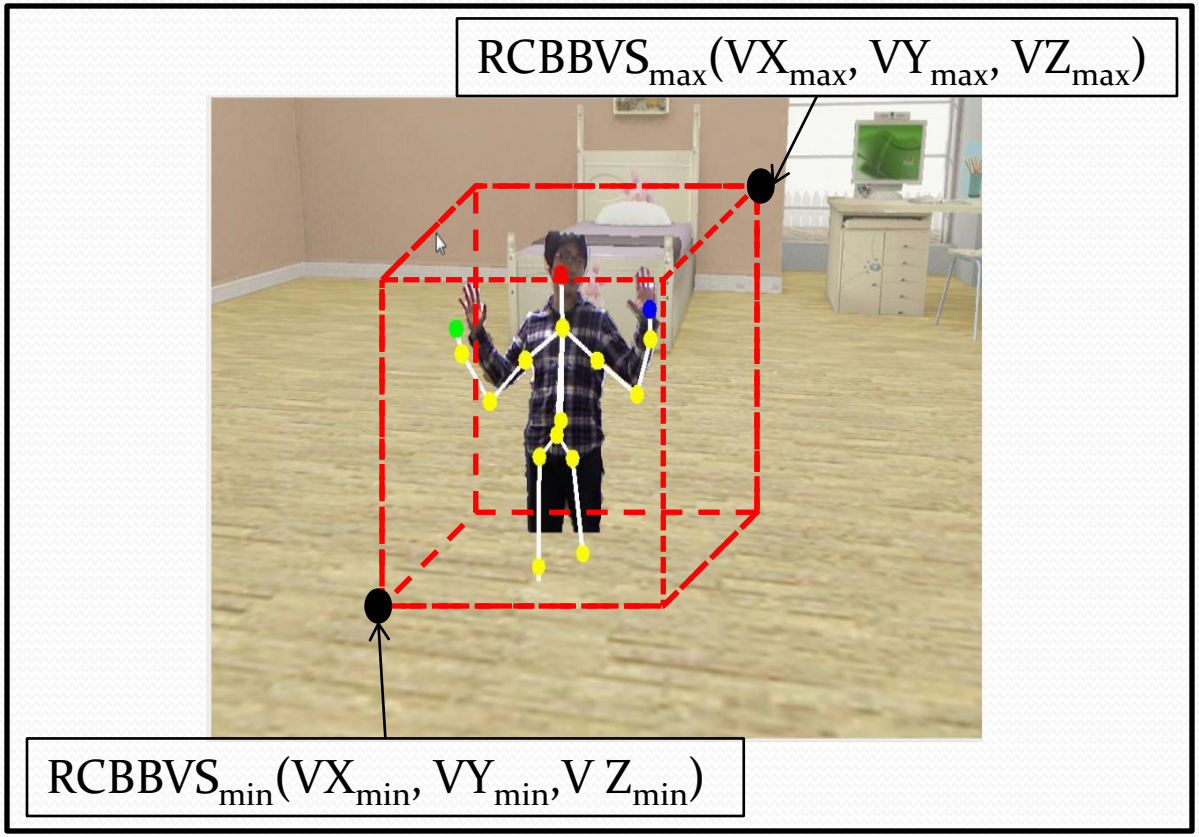
실 캐릭터+
3D 가상 모델

가상공간에서 실 캐릭터의 움직이는 영역 설정 방법

가상공간에서 실 캐릭터의 움직이는 영역 설정 노드

```
RCBBVSNode
{
  SFString              id              ""
  SFString              description     ""
  SFVector[in, out]     startpoint      0 0 0
  SFVector[in, out]     endpoint        1 1 1
  SFString[in]          virtualspace    null    // 가상공간 오브젝트 로드
  SFVector[out]         vsStartpoint    0 0 0
  SFVector[out]         vsEndpoint      1 1 1
}
```

가상공간에서 실 캐릭터의 움직이는 영역 설정(RCBBVS)

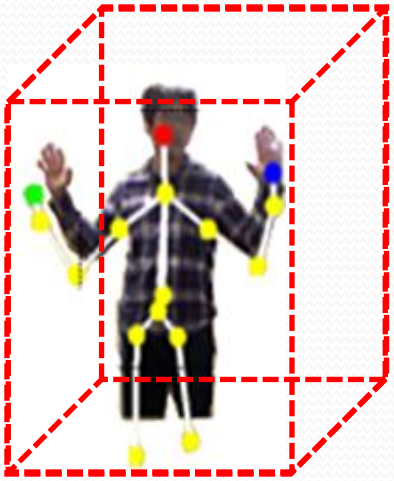


가상공간에서 실 캐릭터의 공간 매핑 방법

가상공간에서 실 캐릭터의 공간 매핑 노드

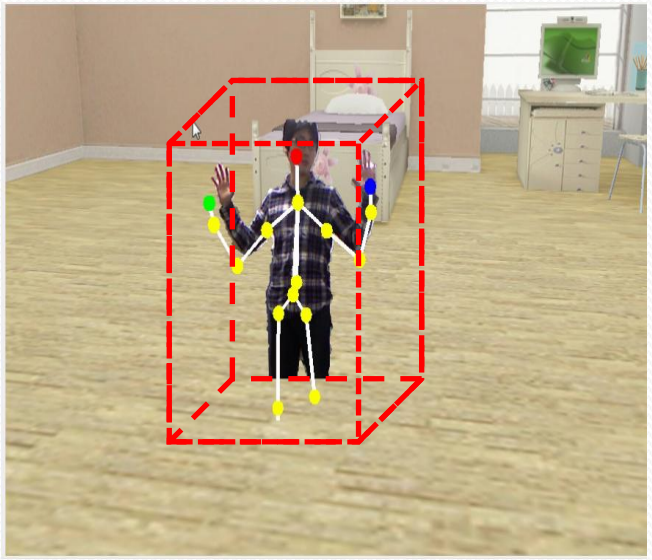
```
RCSpatialMapper
{
  SFString                id              ""
  SFString                description     ""
  RCBBRSNode[in, out]     realspace       ""
  RCBBVSNode[in, out]     virtualspace    ""
  SFVector[in, out]       direction       0 0 1
  SFVector[in, out]       scale           1 1 1   // 크기조절
  SFVector[in, out]       up              0 1 0
  SFString[in, out]       vcObject        Null
}
```

가상공간에서 실 캐릭터 매핑



매핑

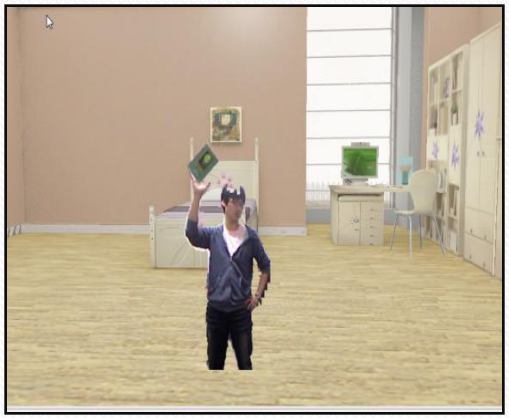
$RZ_t \rightarrow VZ_t$



실공간<realspace>

가상공간< virtualspace >

결과



object 매핑<vcobject>

가상공간에 실캐릭터 매핑

실 캐릭터 표현 예제

실 캐릭터 표현을 위한 일반 카메라 센서

```
<RCSensingDeviceNode id = "cam0" type = "camera" fov="50" framerate="30"> </
RCSensingDeviceNode >

Get two points for a bounding box in real space
<RCBBRSNode id =  "bbrs1" description =  "movable space of real characters in
real space" startpoint=  "0 0 0" endpoint=  "640 480 100"  sensingDevice
=  "cam0"  > </ RCBBRSNode >

Get two points for a bounding box in virtual space
<RCBBVSNode id =  "bbvs1" description =  "movable space of real characters in
Virtual space" startpoint=  "30 30 20" endpoint=  "60 40 10"
virtualspace=  "carobject.3xd"  > </ RCBBVSNode >

<RCSpatialMapper id =  "rcsp1"  realspace =  "bbrs1"  virtualspace=  "bbvs1"
direction=  "0 0 1"  vcObject=  "carModel.xd"  >
</RCSpatialMapper>
```

실 캐릭터 표현 예제(1/2)

실 캐릭터 입력을 위한 깊이 카메라 센서

```
<RCSensingDeviceNode id = "depthcam1" type = "depthcamera" fov="50" framerate="30 " jointTypes = "skullbase" values = "skullbase _x skullbase _y skullbase _z" > </RCSensingDeviceNode >

<RCSensingDeviceNode id = "depthcam1" type = "depthcamera" fov="50" framerate="30 " jointTypes = "l_wrist" values = "l_wrist_x l_wrist _y l_wrist _z" > </RCSensingDeviceNode >
```

Get two points for a bounding box in real space

```
<RCBBRSNode id = "bbrs2" description = "movable space of real characters in real space" startpoint= "0 0 0" endpoint= "640 480 100" sensingDevice = "depthcam1" > </RCBBRSNode >
```

-continue-

실 캐릭터 표현 예제(2/2)

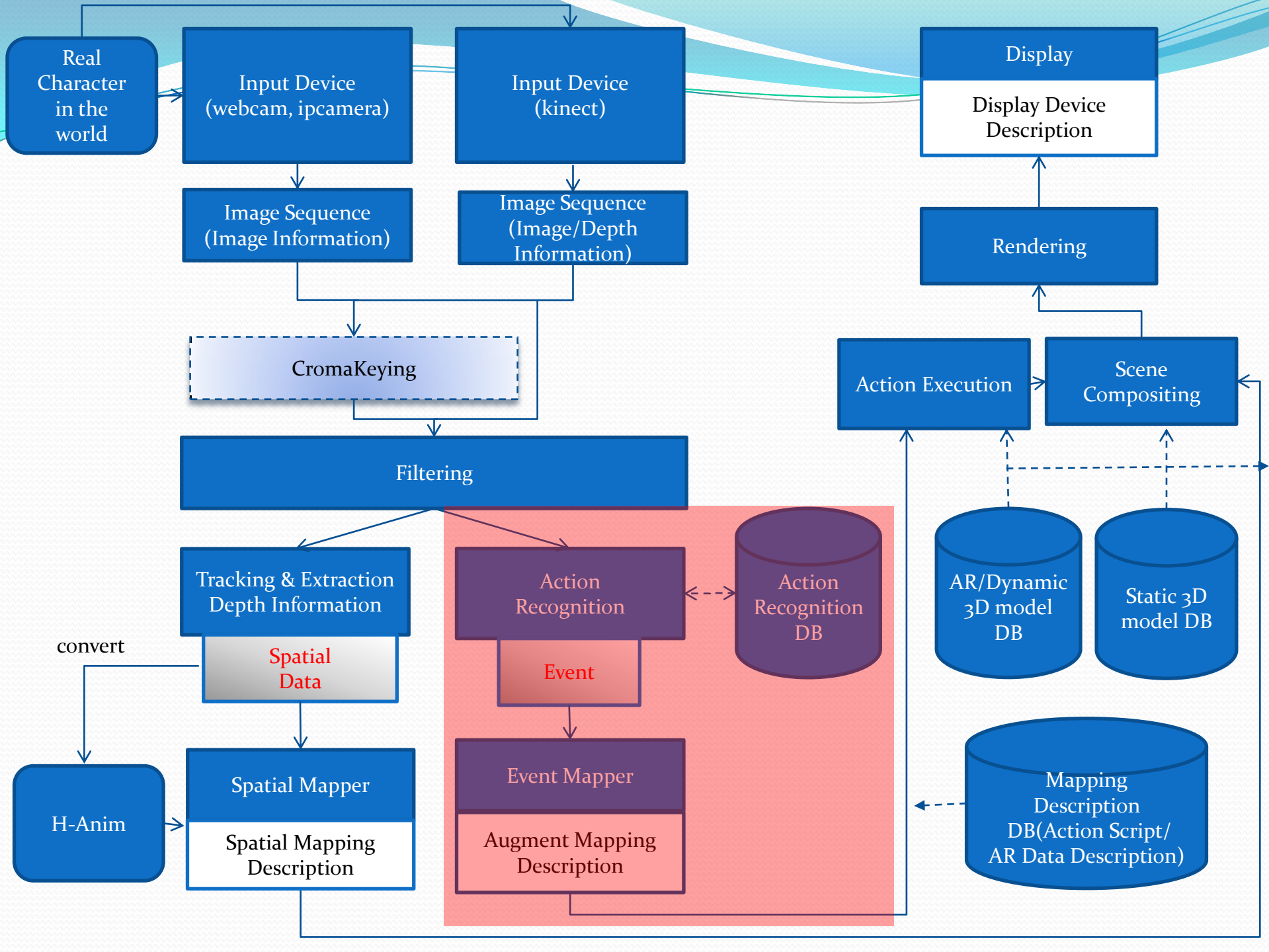
실 캐릭터 표현을 위한 깊이 카메라 센서

Get two points for a bounding box in Virtual space

```
<RCBBVSNode id = "bbvs2" description = "movable space of real characters in Virtual space" startpoint= "30 30 20" endpoint= "60 40 10" virtualspace= "carobject.3xd" > </RCBBVSNode >

<RCSpatialMapper id = "rcsp2" realspace = "bbrs2" virtualspace= "bbvs2" direction= "0 0 1" vcObject= "carModel.xd" >
</RCSpatialMapper>
```

DEMO of Spatial Mapper


Real Character in the world
Input Device (webcam, ipcamera)
Input Device (kinect)
Display
Display Device Description
Image Sequence (Image Information)
Image Sequence (Image/Depth Information)
Rendering
CromaKeying
Action Execution
Scene Compositing
Filtering
Tracking & Extraction Depth Information
Action Recognition
Action Recognition DB
AR/Dynamic 3D model DB
Static 3D model DB
convert
Spatial Data
Event
Event Mapper
H-Anim
Spatial Mapper
Spatial Mapping Description
Augment Mapping Description
Mapping Description DB(Action Script/ AR Data Description)

Action Recognition

Action Types

GESTURE_XXXX_XXXX
– Hand, Foot, Arm, Leg,
Face, Eye, Full body, image
VOICE_XXXX_XXXX
etc

Events

point

image

vector

Action Recognition

Action Types

GESTURE_XXXX_XXXX
– Hand, Foot, Arm, Leg,
Face, Eye, Full body, image
VOICE_XXXX_XXXX
etc

Gesture Events

point

P= (x,y,z)

P2

P1

vector

P1=(x1,y1,z1)

P2=(x2,y2,z2)

image

skeleton

Gesture Events

Posture  point

P= (x,y,z)

vector

P1=(x1,y1,z1)

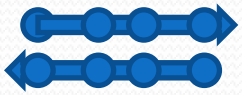
P2=(x2,y2,z2)

image

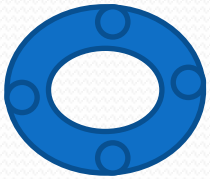
skeleton

Gesture 

X_POS, X_NEG
Y_POS, Y_NEG
Z_POS, Z_NEG



X_WAVE
Y_WAVE
Z_WAVE

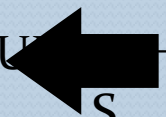


CW(CLOCKWISE)
CCW(COUNTER CLOCKWISE)

Gesture Events Types

Action	Objects	Move	Definition of action types
Gesture	Left hand(LH)	The x positive direction	GESTURE_LH_X_POS
	Right hand(RH)	The x negative direction	GESTURE_RH_X_NEG
	Left foot(LF)	The y positive direction	GESTURE_LF_Y_POS
	Right foot(RF)	The y positive direction	GESTURE_RF_Y_POS
	Left Lower Arm(LLA)		
	Left Arm(LA)		
	Right Lower Arm(RLA)		
	Face(F)		
	Eye(E)		
	Full body(FB)		

Use case for controlling a camera movement by using Gesture Events of Real Characters

	LEFT_HAND	RIGHT_HAND	Functions
Camera control using hand gestures	 GESTURE_LH_X_NEG	-	**Left** (Rotation) Movement of a camera in x−positive direction
	-	 GESTURE_RH_X_POS	**Right** (Rotation) Movement of a camera in x−negative direction
	 GESTURE_LH_Y_POS	 GESTURE_RH_Y_POS	**Up** (Rotation) Movement of a camera in y−positive direction
	 GESTURE_LH_Y_NEG	 GESTURE_RH_Y_NEG	**Down** (Rotation) Movement of a camera in y−negative direction
	 GESTURE_LH_X_NEG	 GESTURE_RH_X_POS	**Zoom In** (Scaling) Decreasing distance between a model center and a camera center
			Zoom Out (Scaling)

Nodes

Nodes for describing events(actions) of real characters

```
RCEvent
{
  SFString [in]     id                  //
  SFString [in]     description         //
  SFString [in]     type                // Event Type
  SFAudio[in]       audio               // captured audio
  SFImage[in]       image               // captured image
  MFString[in       jointTypes          // names of joint positions
  MFVector[in]      values
                    // depth values for the joint positions
  MFString[out]     resValues
}
```

Use Cases of Event Mapper

< RCSensingDeviceNode id = "camo" type = "camera" fov="50" framerate="30"> </device>

<RCEvent id = "event1" description = "controlling position and orientation of camera" image ="input.img" **resValues =**"GESTURE_LH_X_NEG"**>**
send "movement information of a camera in x−positive direction" to camera node
</RCEvent >

<RCEvent id = "event1" description = "controlling position and orientation of camera" image ="input.img" **resValues[0] =**"GESTURE_LH_X_NEG" **resValues[1] =**"GESTURE_RH_X_POS" **>**

send "decreasing distance information between a model center and a camera center" to camera node

</RCEvent >

DEMO of Event Mapper

Control Camera & Objects according to actions of real characters

Conclusion

- NWIP to be submitted to ISO/IEC JTC 1/SC 24/WG 9
  - ARC Reference Modules for Real Character Representation
- More uses cases and implementation
- Documentation

참조문헌

[1]J. S. Jeong, S. A. Kwon, J. A. Park, C. Park, N. H. Baek, K. H. Yoo, Reference Model for Representing Real Characters into the 3D virtual Space, Information Science and Technology 2012 . 1권 1호 346 ~ 348

[2]장효영외, 3차원 공간상의 수신호 인식 시스템에 대한 연구, 전자공학회논문지 2004.5. 제41권 CI편 제3호

[3] 김용완외, 실감형 가상현실 상호작용 기술동향, 2012 Electronics and Telecommunication

[4] Microsoft Corp, KINECT for Windows. (2012)

[5]http://www.web3d.org/realtime-3d/category/pages/h-anim

[6]http://chamadam.tistory.com/5

[7] 김용훈외, 혼합현실기반 이러닝 기술동향, 2009.2 전자통신동향분석 제 24권 제1호

[8] 이재욱, 이명원, 캐릭터 애니메이션 데이터의 H-Anim 기반 정의, 2009.10 정보관학회논문지:컴퓨팅의 실제 및 레터 제 15권 제 10호

[9] 이명원, 휴먼 모션 데이터 표준화, 2008 웹디지털 콘텐츠 표준화 포럼


THANK YOU