FINAL FANTASY XIII's Motion-Controlled Real-Time Automatic Sound Triggering System

Speakers

Yoshinori Tsuchida

Technical Director

Audio Programmer

Tomohiro Yajima

Sound Director

Sound Designer

Motion-Controlled Real-Time Automatic Sound Triggering System:
Abbreviated as “MASTS”

SQUARE ENIX

The Value of Sound Effects

＜Physics-Based　Acoustic Rendering＞

- Add atmospheric depth
- Bring balance to the total of all audio elements
- Allow subtle yet effective expression of character emotions

Expressing Character Emotions Through Footsteps and Rustling of Clothing

- Expressing emotions through dynamics in weak and strong actions
- Expressing emotions through varying speeds
- Sound production in both upper body close-ups and full-body overhead views
- Sound effects from sneaking and quiet footsteps
- Sound effects from the swinging of arms
- Even determination of whether or not sound effects will occur
- Use in gauging character location and distance

My Personal Roadmap to Sound Automation - 1

We first conceived of automation during this period when we worked on the complete expression of emotions through footstep and action sound effects to make up for the game's lack of voices.

Vagrant Story (PlayStation®/2000)

My Personal Roadmap to Sound Automation - 2

We prepared a large number of terrain types and put them in a database so that each terrain could be associated with specific sounds. This, in combination with 3D Panning and Distance Dissipation Algorithms, helped to add realism to the player's experience.

FINAL FANTASY XI
(PlayStation®2,Windows®,Xbox360®/2002～)

SQUARE ENIX

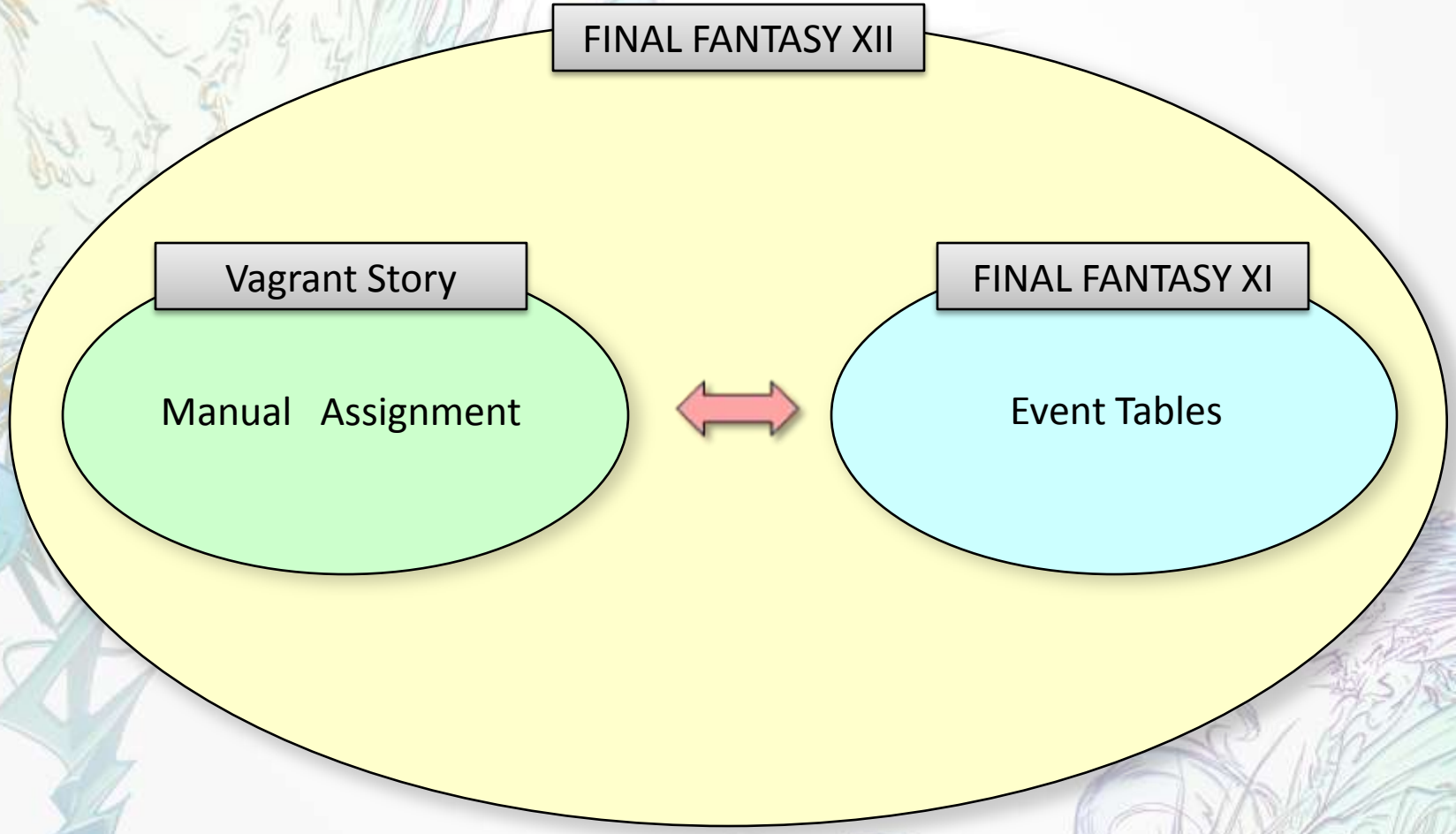
My Personal Roadmap to Sound Automation - 3

We put all of the know-how we had acquired up to this point into practice and used automation for strong and weak footsteps.

FINAL FANTASY XII
(PlayStation®2/2006)

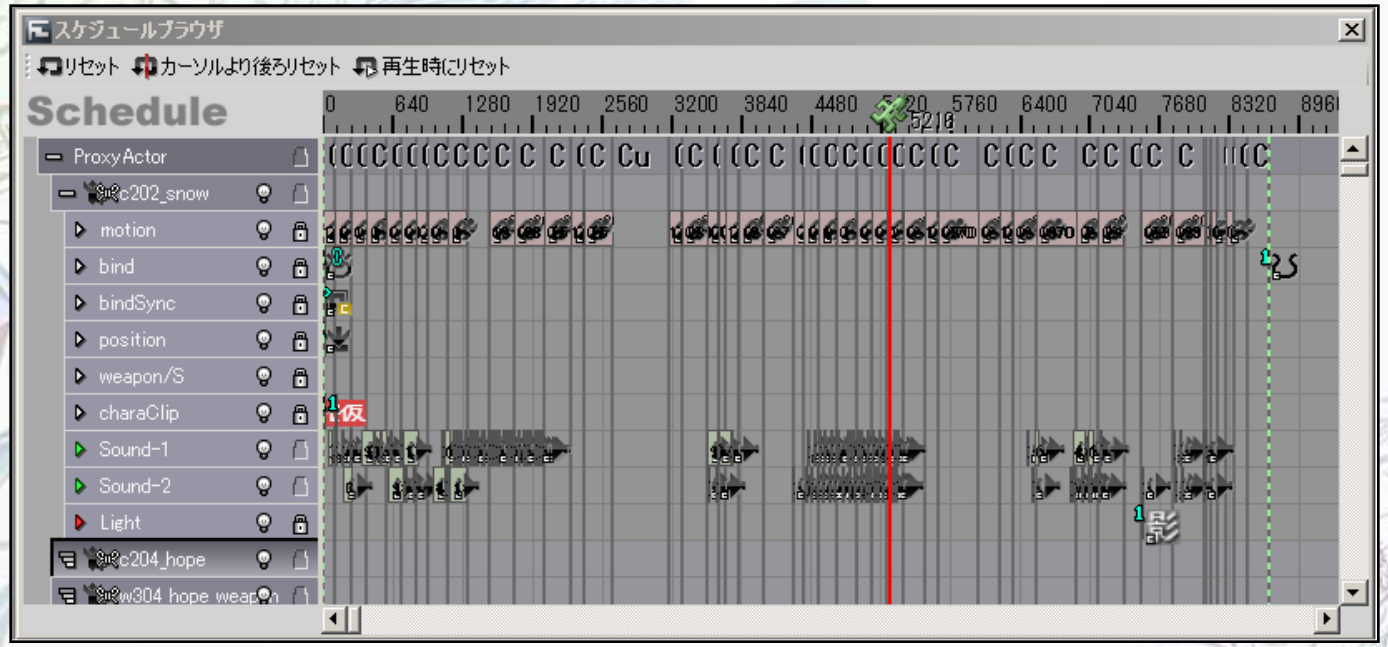
SQUARE ENIX

Benefits and Disadvantages of the Traditional Method



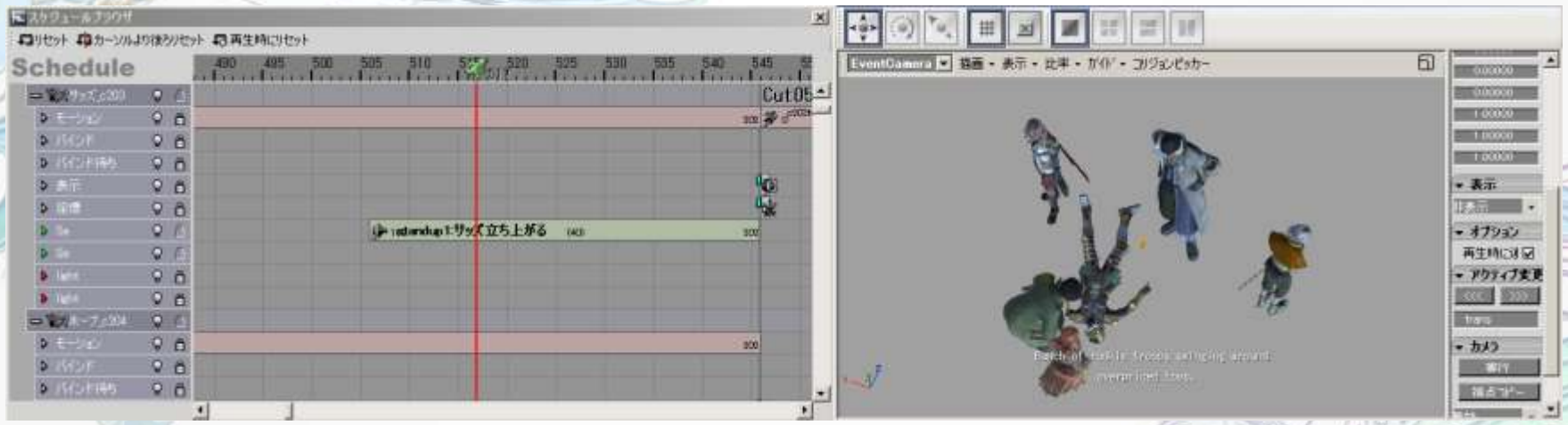
Reasons for Automation: Reason #1

The work costs of designating each footstep sound individually are huge.



Reasons for Automation: Reason #2

If an individual motion or an entire scene needs to be changed, the timing of the sounds will need to be minutely adjusted, taking up far too much time.



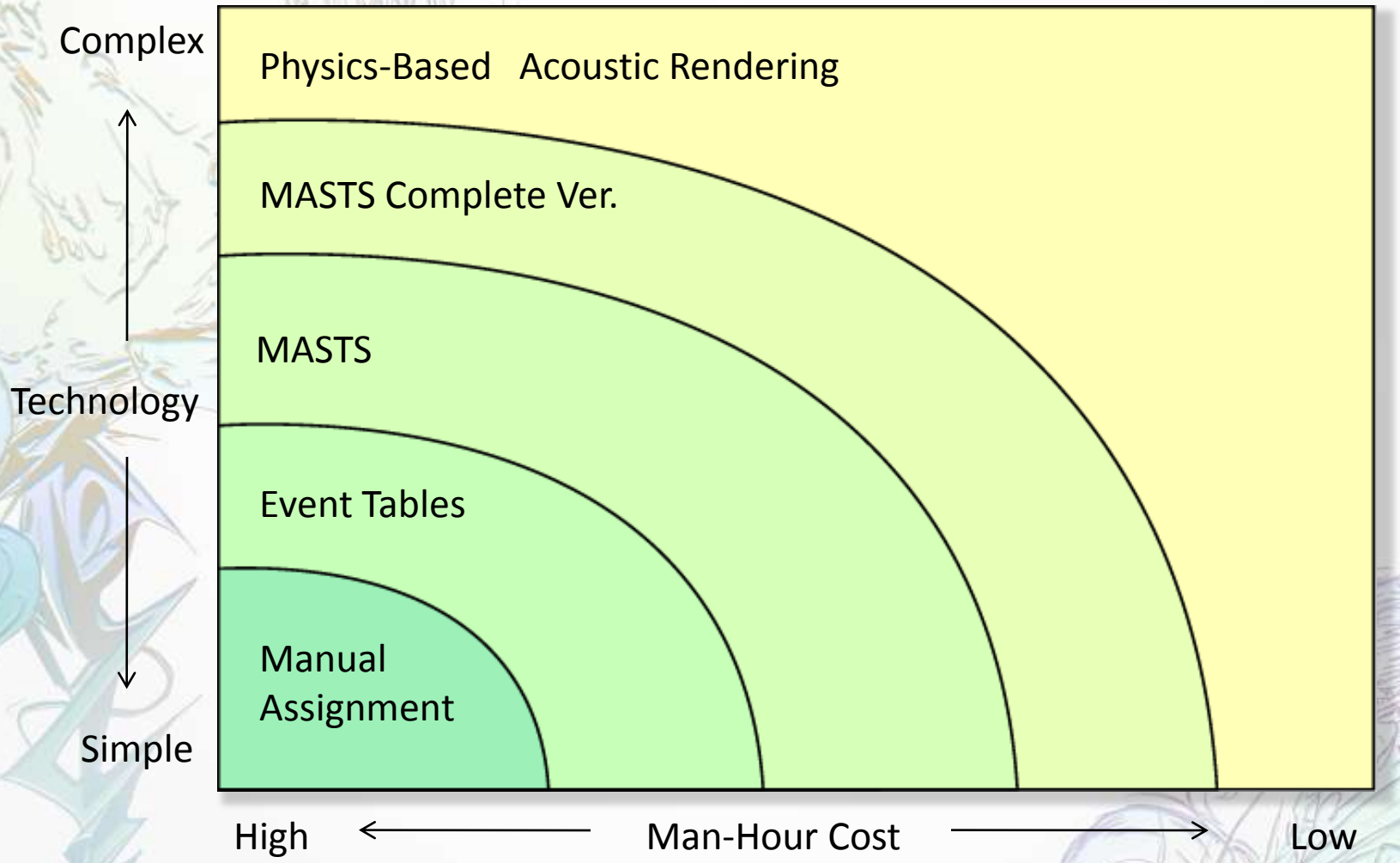
Reasons for Automation: Reason #3

Not good for interactivity.

FINAL FANTASY XIII – Our New Approach

SQUARE ENIX

FINAL FANTASY XIII – Our New Approach



Footstep Data in Past Projects

	Number of footstep sounds in table data	Number of cutscene-specific footstep sounds	Dealing with detailed expressions
Vagrant Story	1568	1400	Possible through manual assignment
FF11	1430	Almost none (around 100)	Very difficult due to the wide variety of possible main characters in an MMORPG
FF12	954	8000	Possible through manual assignment
FF13	5200	None	Dealt with through automation

Moving from Manual Procedures to Automation

＜Original Concept＞

- Gyro-sensors inserted in each joint
- Sound effects produced based on speed
- Extent of actions determined based on gyro location and distance
- Sounds determined based on combination of each action

Challenges of Implementation

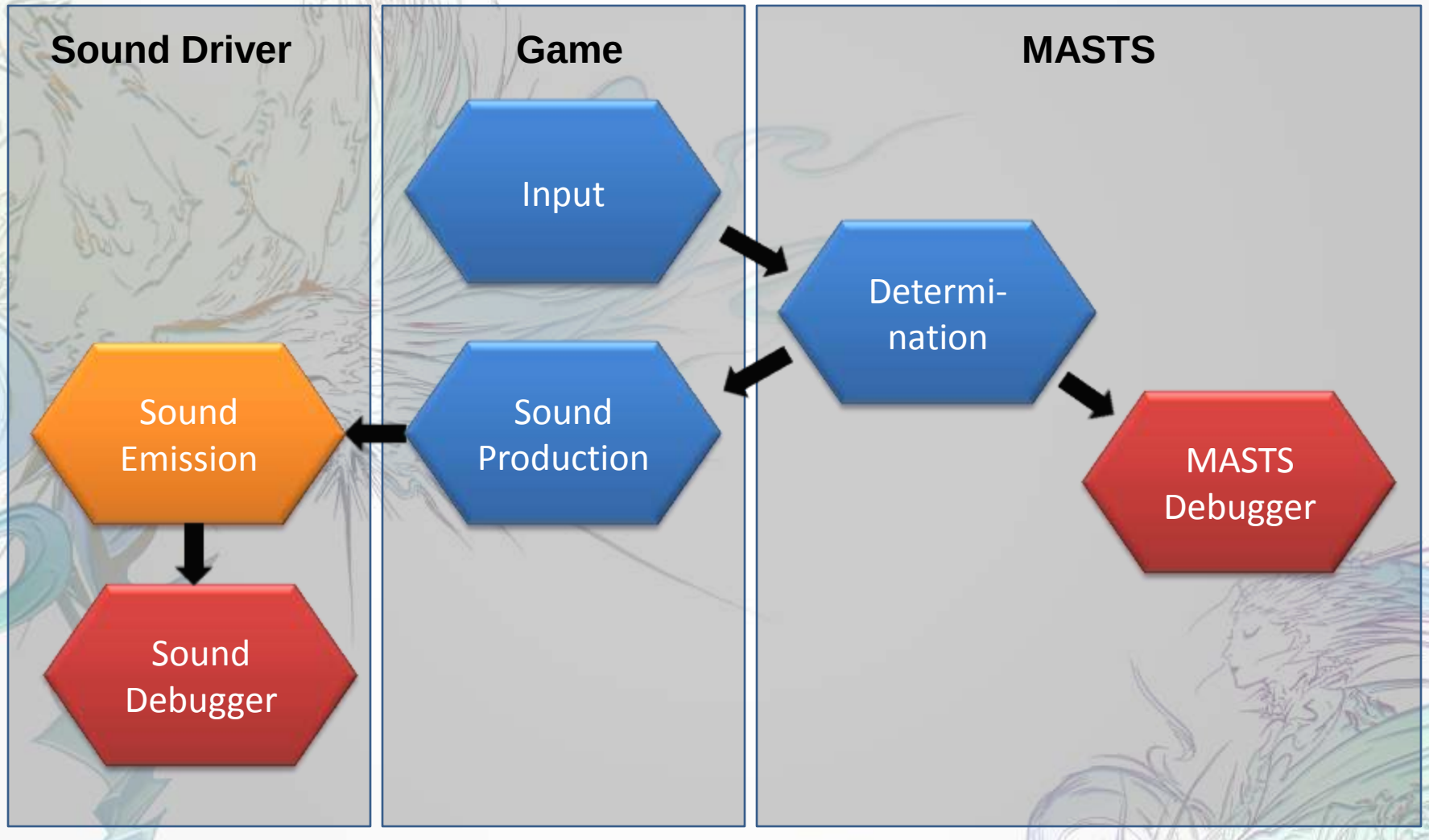
- Need for high-level programmers
  - Programmers with experience and expertise in many various fields
- Coordinating with many other non-sound related teams
  - Always a big challenge
- Creating various support tools

Challenges of Implementation

Sounds like fun! Let's do it!

Let’s have a listen.

Overall Data Flow



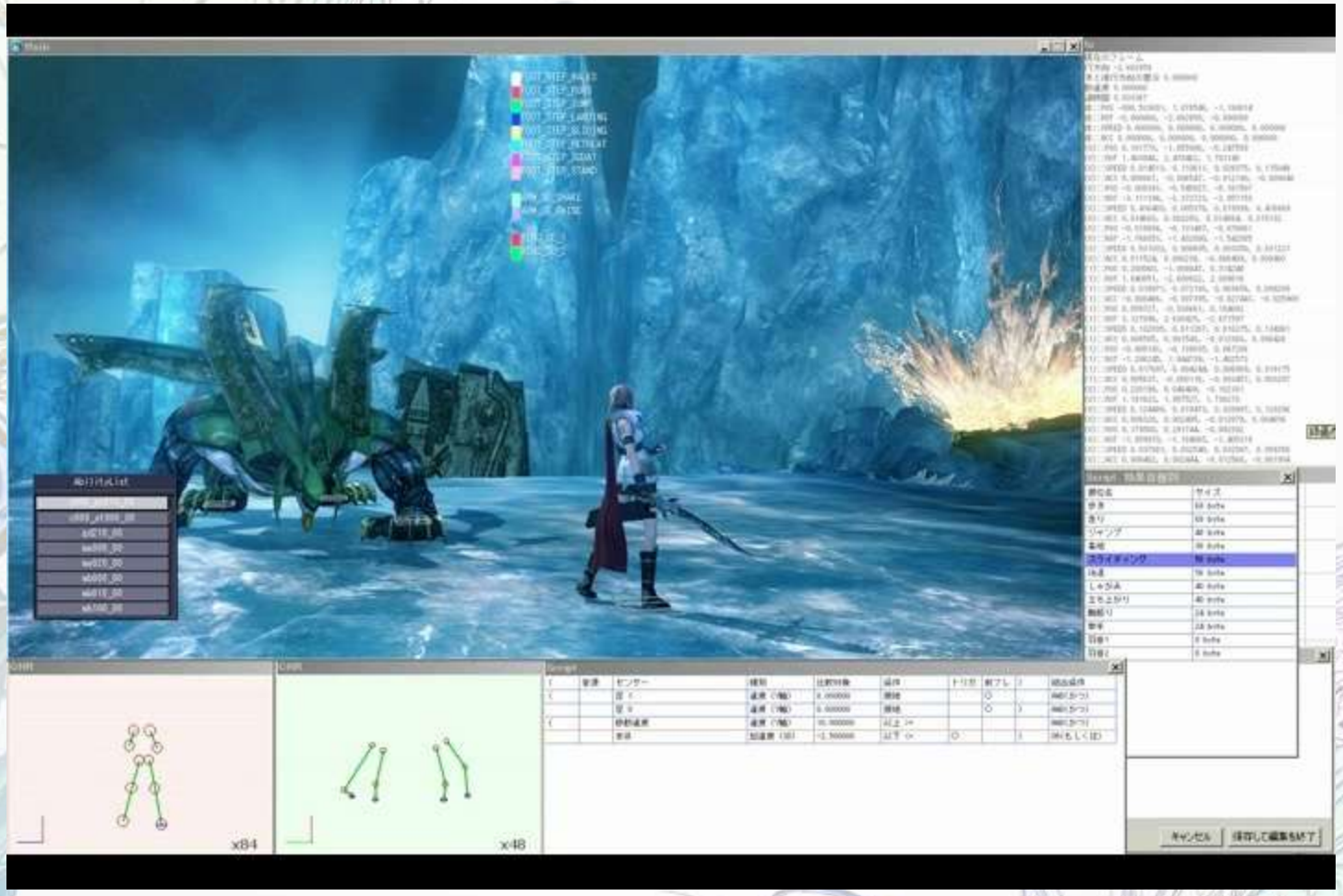
The Core of MASTS

- Three phases:

Input Phase

- Information used in physics calculations:
  - Character's direction, movement speed, etc.
  - Angle and coordinates of character's bones
  - Collision Detection Data
  - Script type (referred to later)
  - Motion type

Input Phase



Data Collection and Analysis

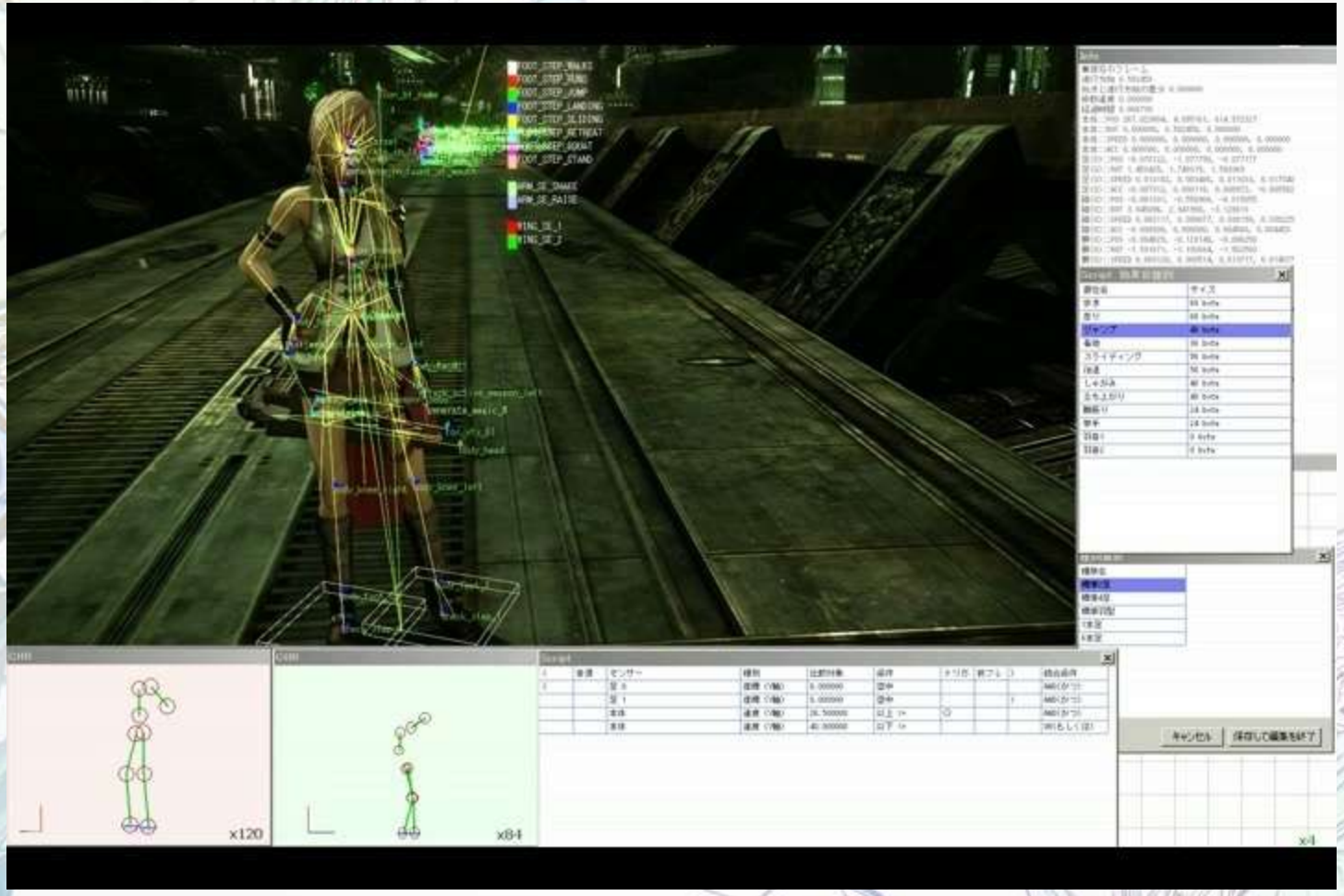
- Examples:
  - Character data
    - Location/position, movement direction, orientation, movement speed
    - Motion classification
    - Time elapsed
  - Set of leg bone data
    - Legs, knees, lower back
  - Set of arm bone data
    - Elbows, shoulders
  - Set of wing bone data
    - Shoulders, wing tips

Coordinates
Angle of rotation
Speed
Acceleration

Determination Phase

- All operations based on inputted data
  - Will a sound be produced or not?
  - What kind of sound will be produced?
    - Landing? Rustling of clothing? Etc.
  - How will the sound be produced?
    - Volume, pitch, etc.

Determination Phase



Sound Production Element Chart

- Rustling of Clothing (4 types)
  - Swinging of arms
  - Raising of arms
  - Crouching, standing up
- Footstep Sounds (6 types)
  - Walking (forward and backward)
  - Running
  - Sliding
  - Jumping (ascension, landing)
- Wing Sounds (2 types)
  - 2 types

Sound Production Phase

- Time for the sound itself! But before that…
  - Character classification, equipment, clothing
  - Terrain type, weather type, location type

"Jumping" sound:

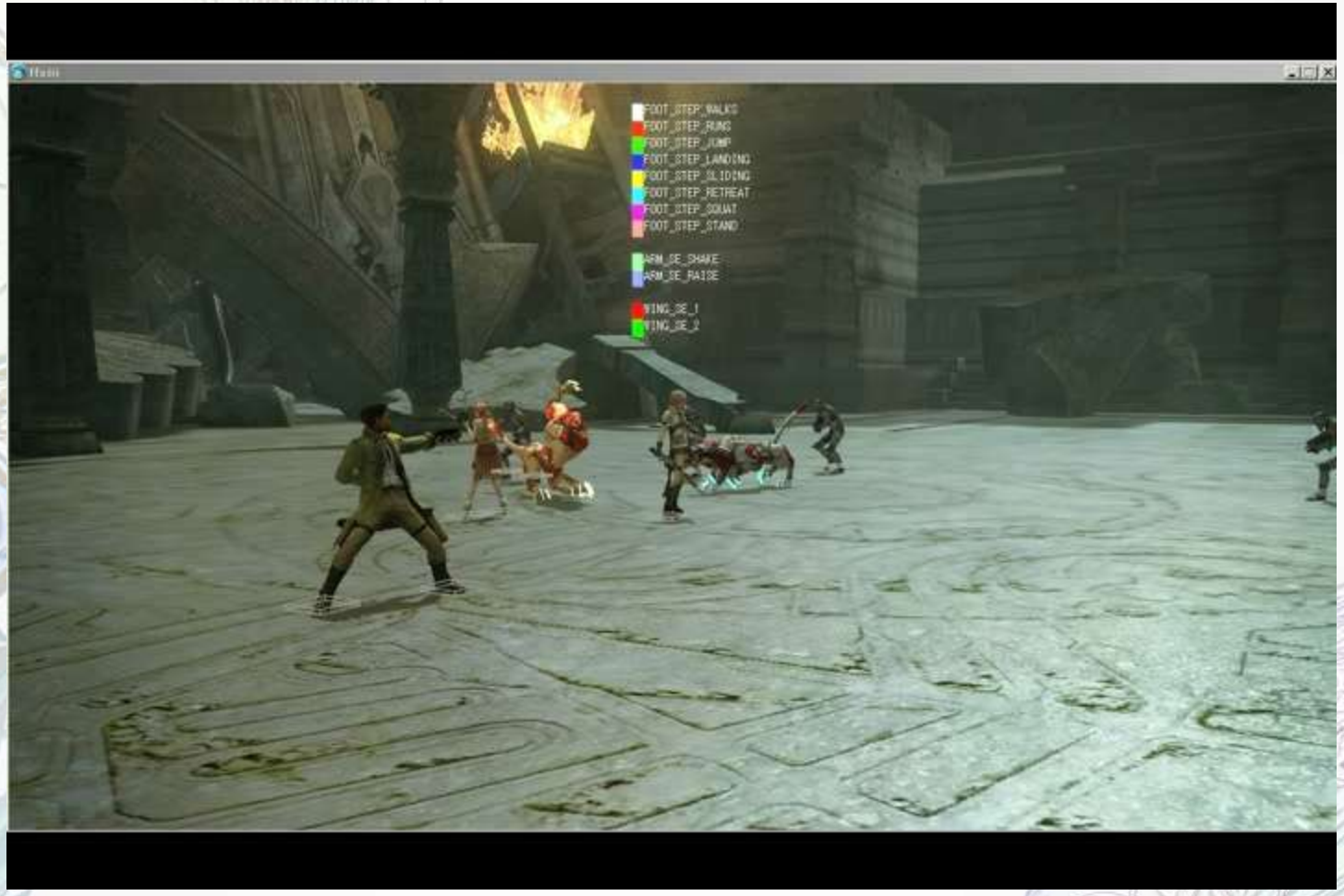
Jumping sound of: main character:

wearing metal boots: on grass:

in the rain

Sound Production Phase

Let’s have a listen.



What? Wait a second!

- The methods for detailed calculations should be different for each type of character and creature.
  - Multi-legged creatures, birds, dragons, caterpillars…
- Are the numerical values minutely adjusted for each character?!

What? Wait a second!

- The methods for detailed calculations should be different for each type of character and creature.
  - Multi-legged creatures, birds, dragons, caterpillars…
- Are the numerical values minutely adjusted for each character?!

Could the team even be using hard-coding?

Specialized Script

- Designates the elements and equations to be used in determination processes
- Gives the parameters and threshold values

Script

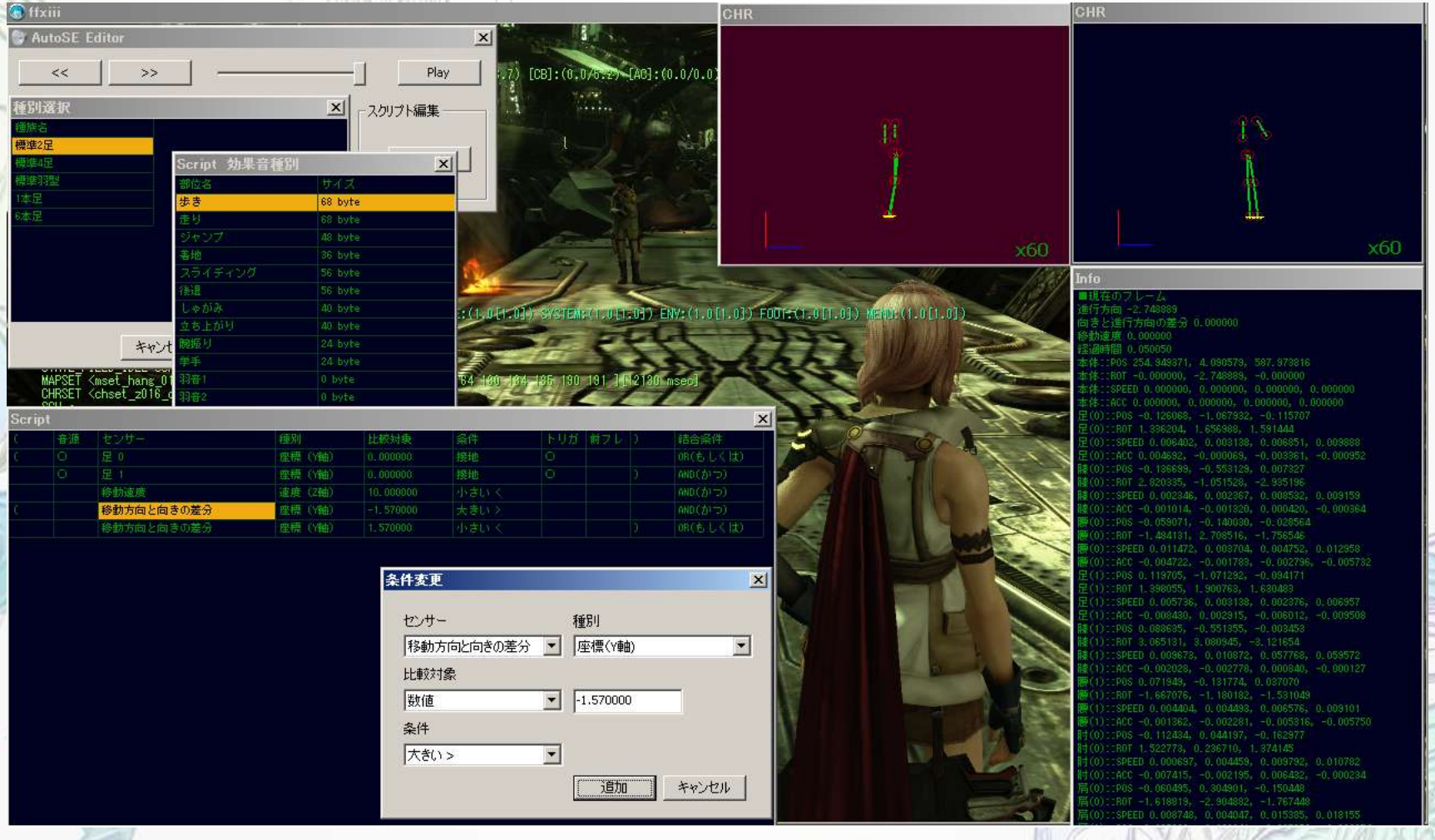
(	音源	センサー	種別	比較対象	条件	トリガ	前フレ	)	結合条件
(	○	足 0	座標（Y軸）	0.000000	接地	○			OR(もしくは)
	○	足 1	座標（Y軸）	0.000000	接地	○			OR(もしくは)
	○	足 2	座標（Y軸）	0.000000	接地	○			OR(もしくは)
	○	足 3	座標（Y軸）	0.000000	接地	○		)	AND(かつ)
		移動速度	速度（Z軸）	10.000000	小さい <				AND(かつ)
(		移動方向と向きの差分	座標（Y軸）	-1.570000	大きい >				AND(かつ)
		移動方向と向きの差分	座標（Y軸）	1.570000	小さい <			)	OR(もしくは)

Specialized Script

- Designates the elements and equations to be used in determination processes
- Gives the parameters and threshold values

- **Complete tool for sound designers!**
- **Data-driven**

Script Tool


ffxiii
AutoSE Editor
<<
>>
Play
種別選択
種族名
標準2足
標準4足
標準羽型
1本足
6本足
スクリプト編集
Script 効果音種別
部位名
サイズ
歩き
68 byte
走り
68 byte
ジャンプ
48 byte
着地
36 byte
スライディング
56 byte
後退
56 byte
しゃがみ
40 byte
立ち上がり
40 byte
腕振り
24 byte
挙手
24 byte
羽音1
0 byte
羽音2
0 byte
CHR
x60
CHR
x60
Info
Script
音源
センサー
種別
比較対象
条件
トリガ
前フレ
結合条件
足 0
座標（Y軸）
0.000000
接地
OR(もしくは)
足 1
座標（Y軸）
0.000000
接地
AND(かつ)
移動速度
速度（Z軸）
10.000000
小さい <
AND(かつ)
移動方向と向きの差分
座標（Y軸）
-1.570000
大きい >
AND(かつ)
移動方向と向きの差分
座標（Y軸）
1.570000
小さい <
OR(もしくは)
条件変更
センサー
移動方向と向きの差分
種別
座標(Y軸)
比較対象
数値
-1.570000
条件
大きい >
追加
キャンセル

Script Details

- Elements
  - Bone designation, input values (16 types)
  - Coordinates, angle, speed, acceleration (4 types)
  - XYZ (3 types)
- Comparative code
  - 14 types (==, >=, ||, etc.)
- Unique specifications
  - Order of calculation priorities, comparisons with previous states and values, etc.

Examples of Script Use #1

- Swinging of Arms
  - (Angle from side of right shoulder > -1.4 & previously was not like this) | |
  - (Angle from side of left shoulder > -1.4 & previously was not like this)

Script

(	音源	センサー	種別	比較対象	条件	トリガ	前フレ	)	結合条件
	○	肩 0	角度（ZY面:横か	-1.400000	大きい >	○			OR(もしくは)
	○	肩 1	角度（ZY面:横か	-1.400000	大きい >	○			OR(もしくは)

Examples of Script Use #2

- Raising of Arms

  (Angle from side of right shoulder > 0.4 & previously was not like this) | |

  (Angle from side of left shoulder > 0.4 & previously was not like this)

Script

(	音源	センサー	種別	比較対象	条件	トリガ	前フレ	)	結合条件
	○	肩 1	角度（ZY面:横か	0.400000	大きい >	○			OR(もしくは)
	○	肩 0	角度（ZY面:横か	0.400000	大きい >	○			OR(もしくは)

Examples of Script Use #3

- Sliding

((Right leg == in contact with terrain &
Left leg == in contact with terrain) &
previously was also like this) &
(Character movement speed >= 10) &
((Character acceleration < -2.5) &
previously was not like this)

Script

(	音源	センサー	種別	比較対象	条件	トリガ	前フレ	)	結合条件
(		足 1	速度（Y軸）	0.000000	接地		○		AND(かつ)
		足 0	速度（Y軸）	0.000000	接地		○	)	AND(かつ)
(		移動速度	速度（Y軸）	10.000000	以上 >=				AND(かつ)
		本体	加速度（3D）	-2.500000	以下 <=	○		)	OR(もしくは)

Examples of Script Use #4

- Crouching

(Right leg == in contact with terrain &
left leg == in contact with terrain) &
((Right knee Y axis position > Right hip) |
(Left knee Y axis position > Left hip)) &
previously was not like this)

Script

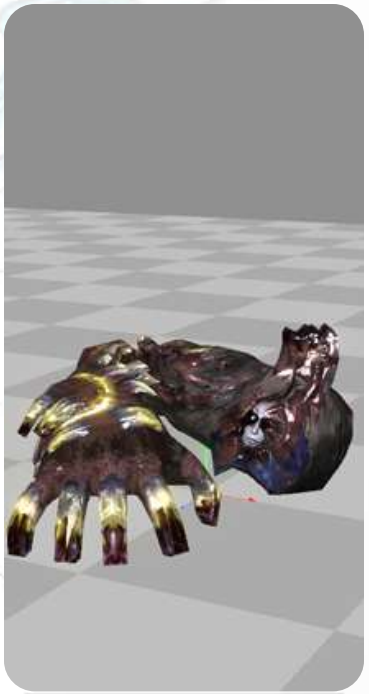
(	音源	センサー	種別	比較対象	条件	トリガ	前フレ	)	結合条件
(		足 0	座標（Y軸）	0.000000	接地				AND(かつ)
		足 1	座標（Y軸）	0.000000	接地			)	AND(かつ)
		膝 0	座標（Y軸）	腰 0	大きい >	○			OR(もしくは)
		膝 1	座標（Y軸）	腰 1	大きい >	○			OR(もしくは)

Problems and Solutions

- Everything A-OK, right? … Wrong!
  - Problems with cutscenes
  - Problems with physics results
  - Unforeseen character actions and numerical values
  - Problems finding the area that needed to be adjusted

Troublesome characters

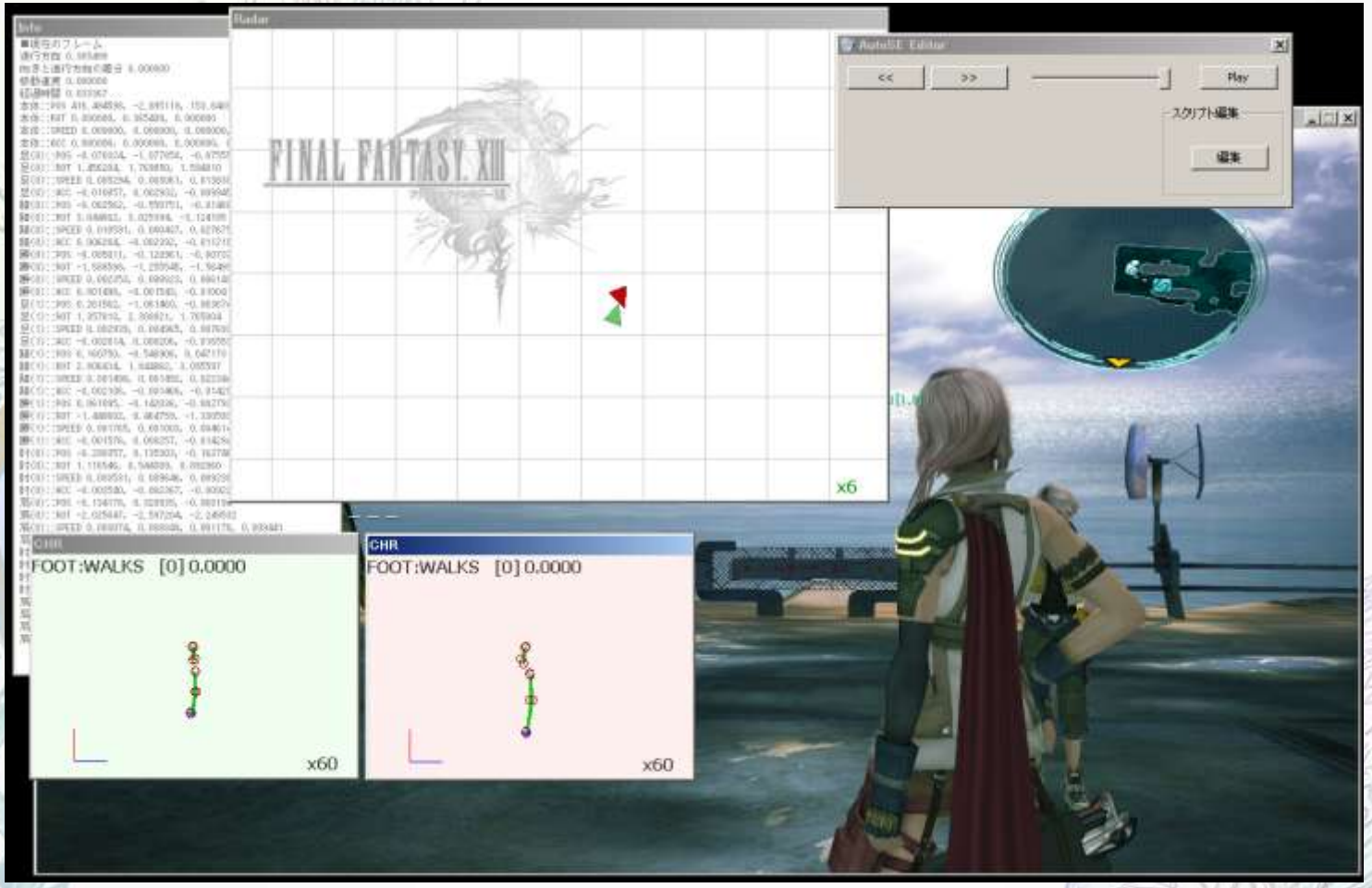




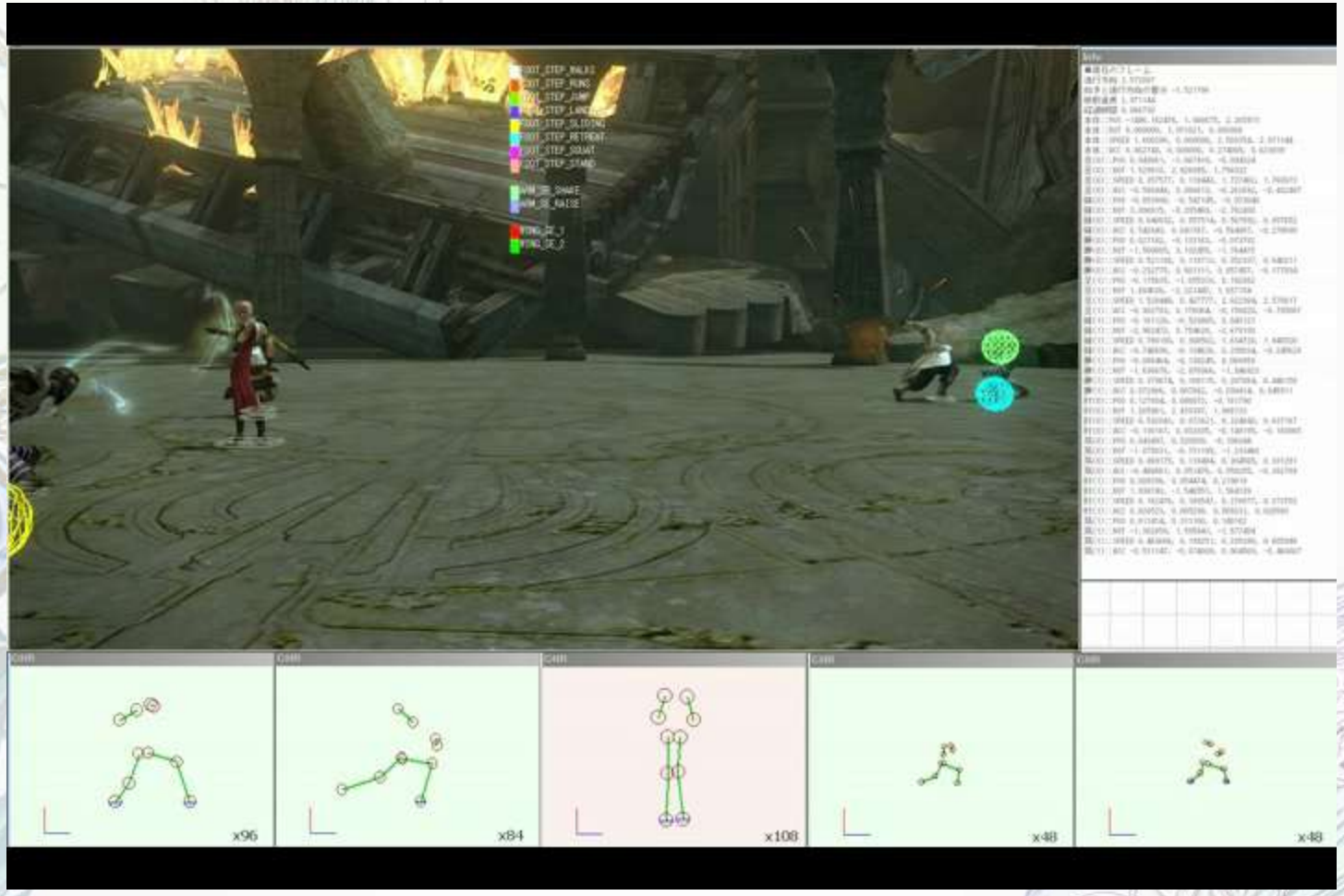

Solutions

- Created a method for automatically detecting irregularities
- Made it possible to use the traditional way of designating sound effects by hand alongside automation
- Created a pre-rendering mechanism

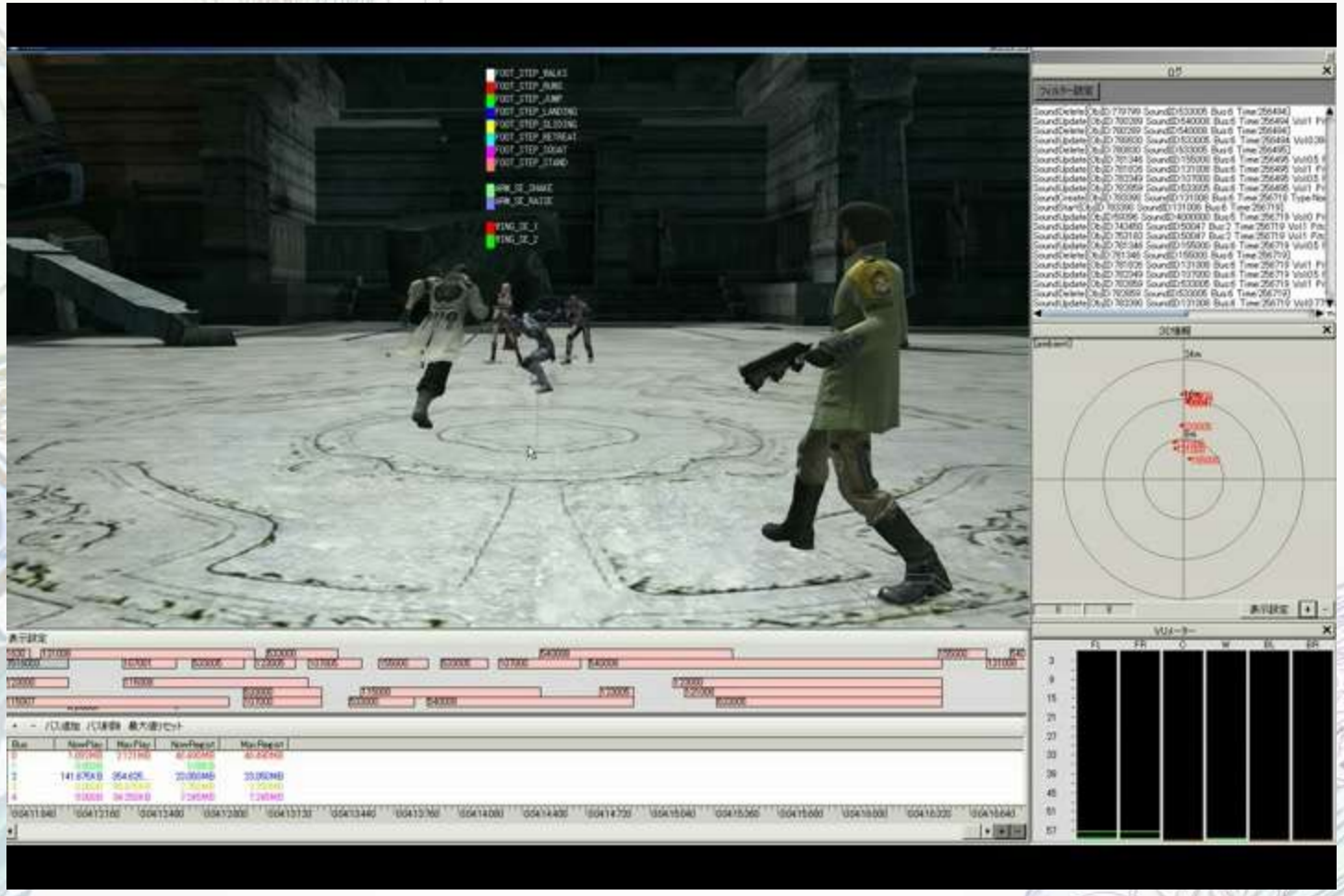
Specialized Debugger



Specialized Debugger



General Debugger



Results

- Work required for manual assignment of sounds was reduced and work required for cutscenes was reduced to one third
- The motion team and planners can freely edit the game without worrying about the sound department!
- Capable of creating minute sound effects based on complicated user actions
- Capable of reacting smoothly to motion changes in real time

Current Projects

- Enrichment of automation patterns in various applications (such as gear and vehicles)
- Auto-adjustments to the appropriation of data templates and bone IDs for new models
- Developing a compact engine that would take the place of table data

Q&A

SQUARE ENIX