

MotionComposer 3.0

Technical Specifications

recent changes:

added section on Field of View
-rw

the section on modes was removed. p50
and re-done according to EE notes.

there are a few open questions (things needing discussion) in red.
pp. 4, 28, 50, 51 (q's top of 16rw).

By ee

Please see CM and TM sections and added blob related OSC messages

Note to Reader: Be respectful of secrecy; there are trade secrets described in this document.
Do not show any part of this document to persons outside of our project. Also, do not discuss the details of our work with persons outside of the project. Thank you.

Table of Contents

MC BASICS.....	5
BRIEF DESCRIPTION.....	5
HISTORY OF THE PRODUCT.....	5
MC TECHNICAL SPECIFICATIONS	8
HARDWARE REQUIREMENTS	8
Chassis	8
Computer.....	10
Tablet Controller.....	10
Sensors.....	10
SOFTWARE REQUIREMENTS.....	13
Software Structure	13
Motion Tracking Module.....	13
Control Module.....	15
Graphical User Interface (GIUs).....	15
MESSAGING PROTOCOLS AMONG MODULES.....	21
Message Addressing Protocol.....	21
Musical Environment(s) Initialization Sequence.....	22
Tracking Module Initialization Sequence.....	23
MESSAGE SEQUENCES.....	24
CONTROL MESSAGES	28
MOVEMENT ALPHABET	29
Introduction	29
The Alphabet Table	30
ALPHABET FEATURES.....	31
OSC Message Reference.....	35
List of Messages:	36
MUSIC ENVIRONMENTS	53
INTRODUCTION	53
Conceptual Requirements (including notes to composers)	53
Note to Composers 1: Causality.....	53
Note to Composers 2: 1- and 2-player schemes.....	54
Note to Composers 3: Initialization and GUI elements.....	54
Note to Composers 4: Evolving mode.....	54
<div>Note to Composers 4: Evolving mode The idea of evolving mode came from a therapist for severely disabled children. He told us his kids are left alone for hours at a time, so that a system that changes by itself over time would be nice. All kinds of changes might can occur, but what comes to mind first is that the instrumentation (or sound bank) changes automatically, over time. The rate of evolution or time span might be “loop after 10mins”, up to “loop after 1hour”, for example. Or random. Or at least which sound comes first could</div>	
.....	54
Note to Composers 5: hL0 and hL3	54
Tracking Modes.....	54
User Modes.....	55
GUI Elements.....	56

ME Technical Requirements	56
TECHNO	56
<i>Note to Composers</i>	56
<div> <p>Note to Composers</p> <ul style="list-style-type: none"> We made a separate document for the music tasks of each ME. Look in the folder “MEWork” and you will see your tasks, and other information, up-to-date. </div>	
.....	56
<i>Description</i>	56
<i>Modes of Use</i>	56
<i>Features</i>	56
<i>Alphabet Used (marked in red)</i>	57
<i>List of Messages (MC 2.0 > 3.0)</i>	59
<i>GUI Elements</i>	61
<i>Misc</i>	62
<i>Future market?</i>	62
FIELDS	62
<div> <p>Note to Composers</p> <ul style="list-style-type: none"> We made a separate document for the music tasks of each ME. Look in the folder “MEWork” and you will see your tasks, and other information, up-to-date. </div>	
.....	62
<i>Note to Composers</i>	62
<i>Description</i>	62
<i>Features</i>	62
<i>Alphabet Used (marked in red)</i>	62
<i>List of Messages Compared in MC 2.0 vs. 3.0</i>	64
<i>GUI Elements</i>	66
TONALITY	66
<i>Note to Composers</i>	67
<i>Description</i>	67
<i>Features</i>	67
<i>Alphabet used (marked in red)</i>	67
<i>List of Messages Compared in MC 2.0 vs 3.0</i>	69
<i>GUI Elements</i>	71
DRUMS	72
<i>Note to Composers</i>	72
<i>Introduction</i>	72
<i>Description of Player Control Features</i>	72
<i>Alphabet Used (marked in red)</i>	72
<i>List of Messages Compared in MC 2.0 vs 3.0</i>	74
<i>GUI</i>	76
PARTICLES.....	76
<i>Note to Composers</i>	76
<div> <p>Note to Composers</p> <ul style="list-style-type: none"> We made a separate document for the music tasks of each ME. Look in the folder “MEWork” and you will see your tasks, and other information, up-to-date. </div>	
.....	76
<i>List of Messages Compared in MC 2.0 vs 3.0</i>	78
<i>GUI</i>	80

IMPORT-YOUR-OWN-MUSIC	81
<i>Note to Composers.....</i>	<i>81</i>

Note to Composers

- We made a separate document for the music tasks of each ME. Look in the folder “MEWork” and you will see your tasks, and other information, up-to-date.

.....	81
<i>Importer (“Import-Your-Own-Music”, etc. we have not found a title yet)</i>	<i>81</i>
<i>Phase I Implementation.....</i>	<i>81</i>
<i>GUI Elements.....</i>	<i>83</i>
<i>Phase II Implementation</i>	<i>84</i>
<i>GUI Elements.....</i>	<i>86</i>
<i>List of Messages Compared in MC 2.0 vs 3.0</i>	<i>87</i>
ADDITIONAL PRODUCT POSSIBILITIES.....	89
<i>Description.....</i>	<i>89</i>
<i>MC PRO-VERSION.....</i>	<i>89</i>
<i>MC MUSICIANS’ VERSION</i>	<i>89</i>

MC Basics

Brief Description

MotionComposer (MC) is a stand-alone device that converts human movement into music. Although other users and markets are envisioned, the MC is being developed primarily for persons with different abilities. Its uniqueness as a product lies in its motion tracking and mapping scheme, its high-quality music and its ease-of-use. These characteristics combine to make the device useful for therapy, education and artistic settings.

Basic Functional Diagram:

moving person → video-based motion tracking ::
:: mapping tracking parameters to sound parameters ::
:: sound generation → music

Terms:

The MotionComposer has two kinds of users:

The person using the computer controls = therapist = researcher = operator

The person moving in front of the camera = patient = subject = player

In this document, we use “operator” and “player”. When we use the word “user”, it refers to either one.

“Modes of Use”. Different musical environments offer different “player paradigms”. There are 2 kinds:

- “Tracking Modes” include: 1-Player, 2-Player and Zones (maybe “Fields”) Zones could also be combined with 1- or 2-Player mode, though in practice, for user- clarity (and other reasons) this rarely occurs.
- “Player Modes” include “In-space” and “In-place”, and other options, but these do not effect the tracking, only the mapping and music. (maybe we need a better naming... see also p50)

“Blob” is the area of the video image defined by a human form, recognized by the Motion Tracking Module. It is often called a “mask” since it defines an area of interest in the video frame.

Abbreviations:

MC – MotionComposer

MA – Movement Alphabet

TM – Motion tracking software module

ME – Music Environment, the software module that generates the music

GUI – Graphical user interface software module

CM – Central control module

RCM – Remote Control Module

UAM – User Administration Module

MC-Pro – The version of the product that allows access the user access to the OSC streams, etc.

MC-Musicians’ – The version of the product with extended Tonality functionality, for music teachers

History of the Product

MC started in 2011 as a start-up (funded by the Germany Ministry of Technology and administered through Bauhaus University). The current version, the MC2.0 (pictured below), was released in June 2015, and, at the time of this writing, is still being sold as a commercial product by IMM electronics GmbH. Ca. 15 MC2.0’s have been sold, mostly to research and educational institutions as well as centers for persons with different abilities. These “test customers” give us use-case experience, as do the over 50 workshops we have conducted. We consult with therapists and other outside experts. A number of papers have been published about the MotionComposer and its applications (<http://motioncomposer.de/full-list-of-publications>).



The MC2.0's Spo.comm Computer and Sensebox. Manufactured by IMM Group, 2012-present.

MC2.0 Description

The MC2.0 consists of a Windows7 desktop computer with its peripherals (monitor, keyboard, mouse) , plus a separate sensor box that includes an ASUS XTION TOF (Time of Flight) sensor (essentially a Kinect) together with a CCD video camera. The Xtion connects to the computer via USB, while the camera uses GigE Ethernet. The images from both sensors are compiled in the custom EyesWeb (Infomus.org) software. Motion tracking, central control and GUI are all written in EyesWeb. One of six musical environments (MEs) is chosen by the user. The ME runs music software (in either Puredata, Csound or Supercollider) in the background.

While it has served us well for workshops, it is not really a satisfying product. Here are some of the issues:

Issue	MC2.0 - problems	MC3.0 - solutions
ease-of-use: hardware	too many components: computer, sense-box, mouse, keyboard, screen, 8 cables to plug in, etc. This was a major problem to many therapists.	all-in-one chassis, wireless controller
ease-of-use: GUI	instruction or training is helpful (in some cases, necessary) Some parts confusing and redundant.	one button play, beautiful intuitive graphical user interface
robustness under different lighting and room conditions	has problems when sunlight is present or with theater lighting (due to infrared). tends to loose person in wheelchair when therapist walks out of the camera. lacks auto aperture.	robust under different lighting conditions "active user" identification and marking auto-aperture
Intuitiveness For both operator and player	Many users are slow to recognize the causal relationship mapping is not always intuitive GUI use of tabs is confusing. GUI has bugs	A visualization of the sound, such as a light inside the chassis, is triggered along with the sound. This strengthens the causal relationship. Improved mapping Tablet controller (touch screen) with completely new GUI
Therapeutically useful with groups	Social interaction is an important aspect of therapy and with the exception of the Fields environment, the MC2.0 is 1-person at a time	2-player mode available for all environments
musicality	some environments are rarely used (are not enjoyed as much) some patients only respond to certain songs (specific to the user) the variety of music offered is too limited "bed" modes (now called "Zones" mode) are sometimes missing or poorly implemented. This mode is very important. techno is popular, but musically limited.	one environment is being removed there is a new ME for importing any song More sound banks and songs Improvements to Zones-only modes. Expansion of "Techno"

Target User Groups

The target user groups for MC 3.0 are

- Persons with disabilities and their therapists

- Severely disabled – persons with very limited movement and/or communication ability
- Educative use – teachers use it with children to develop their bodily control and other skills
- Elderly people – life-quality activities. Elderly people use it for entertainment and body practices
- Anyone – entertainment. People may use for pure entertainment purposes, events, parties, etc.

The target user groups for MC Pro-Version (see the chapter on “Additional Product Possibilities”):

- Artist-Developers – artists can use MC as a motion tracking tool
- Gaming – people may develop gaming environments
- Researchers, universities and research centers – use MC as a research tool
- With special software can have spin-off uses, for example as a testing and therapy tool for CI (cochlear implant) patients, in which variable frequency ranges are needed. It could aid persons who are learning to hear for the first time, to associate their new sensor experiences with their bodies. (In discussion).

The target user groups for MC Musicians’-Version (see the chapter on “Additional Product Possibilities”):

- Music teachers -- with its programmable chords, music teachers have expressed interest in using it to teach music fundamentals to children and young adults.
- Professional musicians (Stevie Wonder expressed interest in using it).

User Experience

The MC is placed on a table with at least a few meters of free space in front of it. It is attached to loud speakers. There are two kinds of users: players and operators.

Player Experience

The player moves and hears sounds or music corresponding to their movement. While this can occur in many different ways, all have one thing in common: The activity should encourage itself – that is, once you start, you feel like continuing, so that one movement, leads to the next, and the next. This encouragement-feedback-loop is supported in two ways:

Cognitive Processes

The player in this case understands the causal relationship between their body, and the sounds they hear. Skill-building, what is known as hard skills, takes place as the user learns to control their body to achieve desired effects. In this sense, the MC is like a musical instrument.

Intuitive Processes

Another aspect of the MC is intuitive. Dancing to music involves a blurring of the senses (known as synesthesia). The pop song says, “I’ve got the music in me”, because this is how it feels: the music seems to come from our body -- or, as Nietzsche said, “We hear music with our muscles”. This phenomenon is practically universal and has important implications for therapy. I.e. even if the patient does not understand how s/he is effecting the music, they can still be encouraged and enlivened through the interactive technology.

While cognitive and intuitive processes are of course very different, our concept of the MC says: they are equally important. This issue is discussed in THE MOVEMENT ALPHABET section.

Operator experience

The tablet is the “control center” for the experience. Its priorities are:

- Minimal set-up time
- Neither visually, nor aurally distracting
- Simple operation

The tablet is used to:

- Check the video image, for example to be sure the player is the picture
- Select the me
- Select 1 or 2 players (also, de-select “active” players)
- Adjust the volume
- Adjust the sensitivity – how much movement is required to achieve results
- Change sound bank, add accent choices, etc. When me provides option

- See errors such as “too close to camera”, “too far from camera”

Additional tablet controls under discussion include:

- possibility to “play along with” the player, by tapping out additional sounds
- in Zone-Tracking mode, manipulate the zones (perhaps pro-version)
- possibility to monitor the player (for example, to observe the Player from another location via video image)
- possibility to collect data on the player’s level of activity, degree of limb extension, etc. (see MUSICAL ENVIRONMENTS – BASICS: E), I).

MC Technical Specifications

Hardware Requirements

The MC hardware consists of two physically-independent units:

- a chassis, containing sensors, and computer
- a tablet controller

The user needs to supply an external speaker system.

Chassis

The BOX should contain:

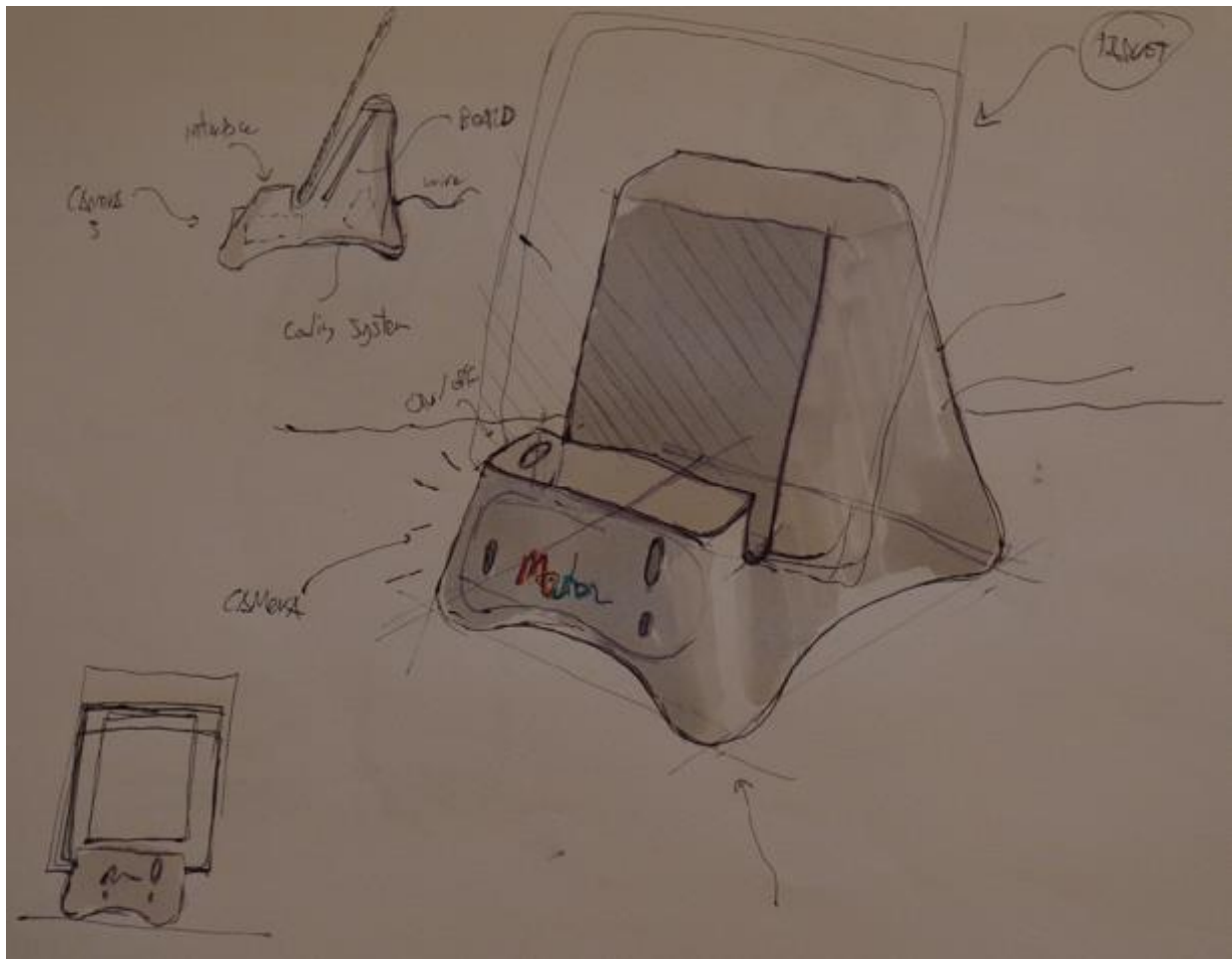
- a mother board standard (intel) micro ATX, sizes: 9.6 × 9.6 in (244 × 244 mm)
(OR may be a mini ITX (150x 150 mm, we will know this soon)
- PCIe 2 (or 4) port Power Ethernet Card
- power supply
- Hard disk or SSD Disk
- Two cameras ca. 15 cm apart; cameras’ dimensions are 42 mm x 29 mm x 29 mm (this is quite certain: Basler Aca 1300-60gm) and lenses on cameras are ca. 23 mm x Ø 24 mm (TAM 12VM412ASIR, or similar. the lens has not yet been chosen)
- Circuit for LED driver
- LEDs
- WiFi capability
- Cooling
- Quiet
- CE certification
- Possibly a space for inserting a TABLET (not decided, yet but u.U. something between ca. 9’ to 7’).
(see drawing)

The BOX should provide access:

- Openly to
 - power button
 - standard jack for power cable
 - Audio output - standard mini jack (“mini-klinke”)
 - 2 USB ports – for Importer ME, and also for Pro-version features
 - HDMI – for pro-version features
 - LED light visibility – through a semi-transparent chassis material
- hidden to
 - Standard PC I/O
 - Ethernet ports

Design Considerations:

- The box should be robust (Behindertengerecht).
- It “lights up” when music is played, so the box needs to be semi-transparent. This lighting should be able to be bright, or dim, or turned off (via software). Not fancy: simple white light.



Additional Chassis Issues (under discussion)

- mountable on a tripod (to be discussed)
- indicator lights, visible to the user.
- to show that it is "on"
- to show when a Player is present, in the correct position, and/or active
- show when movement is causing sounds. This "visual cue" is often helpful for a deaf person
- warn when the player is too close to the camera (the warning should not be too strong, since "being too close" is a matter of how it is being used. Normally, the idea would be to warn when "the whole body, with arms stretched overhead cannot be seen"), but, for someone in a small room, who is only using one body part to play music, they may want to be closer than that). Thus, an indicator light (on the chassis, or GUI) sounds useful, but not a large blinking red light, etc.

While the normal way to I/O with the MC is using the tablet controller, some indicator lights may be desirable. A power light is normal, but the others could just be confusing and redundant to the tablet.

* We don't think a dedicated hardware platform (like Nvidia Jetson board) is a good choice. We are not sure about the Nvidia roadmap to this product, the availability and so on ...

* We suggest a high available standard platform for the mainboard (e. g. μ ATX). It should be possible to add a Nvidia graphic card, but if this option will be necessary is not clear at the moment.

* Every interface (e. g. USB, HDMI ...) that is passed out for service reasons or something else should be connected to the MC housing with a short cable and not directly from the used board. So the type of board can be changed without changing the layout of the MC box.

* For the current idea - Basler Ethernet cameras - the board should also have another PCIe slot for a network card. A good technical solution is a 2-port Power over Ethernet card (or a 4-port version is available for the same price, even sometimes cheaper). So only one cable for each camera is needed to get it working. There must be a power connector for the network card

Computer

- motherboard must have the necessary computational power
- have the option for external video processor if needed (NVidia, etc) – (this is under discussion)
- has a compact size
- has a quiet cooling system
- be single board
- be running linux; accordingly all software to be ported to linux
- have high-quality sound processing capacities
- have wifi capability
- have appropriate i/o including Ethernet, USB and video
- power supply included in box

Tablet Controller

The graphical user interface (GUI) of the system will be available through a tablet that communicates with the computer (MC3.0) over Wi-Fi. When turned on, the Tablet will automatically run a default application, namely, the MC3.0 GUI APP. The Tablet will be shipped with the MC3.0 and will be an Android device.

Client may use other compatible iOS or Android devices as well once the application downloaded and installed. Therefore both version of the MC3.0 GUI APP application that run on Android and iOS devices could be available. The MC3.0 GUI will be optimized for the chosen (ca.) 9" device (like Mini iPad and similar size android tablets), but could also work on smaller, mobile devices.

Sensors

Sensor Type

Even though the MC2.0 uses a Kinect-type sensor, we decided not to use one in the 3.0, due to:

- problems with IR-based sensors (particularly in sunny rooms, or with theater stage lighting)
- The bus (usb and similar) causes unavoidable latency.
- The camera resolution is too low to see small finger, face and eye movements. (the MC2.0 uses an additional CCD)
- Acquiring rights to use the device is difficult. Extracting the sensor from its chassis leaves us open to reverse engineering lawsuits.
- The field-of-view is somewhat smaller than ideal for the typical therapy room.

Instead, the MC3.0 will use true stereo-vision technology, based on two symmetrical CMOS sensors: Basler acA1300-60gm: monochrome, 1280 x 1024, 60 fps, 5.3 x 5.3 pixel size in micrometers, 1/1.8" chip size, e2v CMOS EV76C560 global shutter.

Image Resolution

We need to be able to register small finger movements (for example, a quick 1 cm movement of one finger) from anywhere in the active area, and eye movements and blinks from 2 meters away from the camera. In our tests, this translated to ca. 1280 x 1024 pixel resolution with a 1/1.8" chip size.

Lens

Since the user will have no access to the cameras, the lenses can be fixed. The aperture needs to be sufficiently open to allow the MC to work in low room-lighting conditions, such as might be found in a typical therapy room. An auto-aperture feature will be used to adjust exposure. Focus should be sharp at a 5-meter distance (and not sharp when the player is close up).

Field-of-View

The field of view, should be ca. 5 meters width at ca. 5 meters distance from camera.

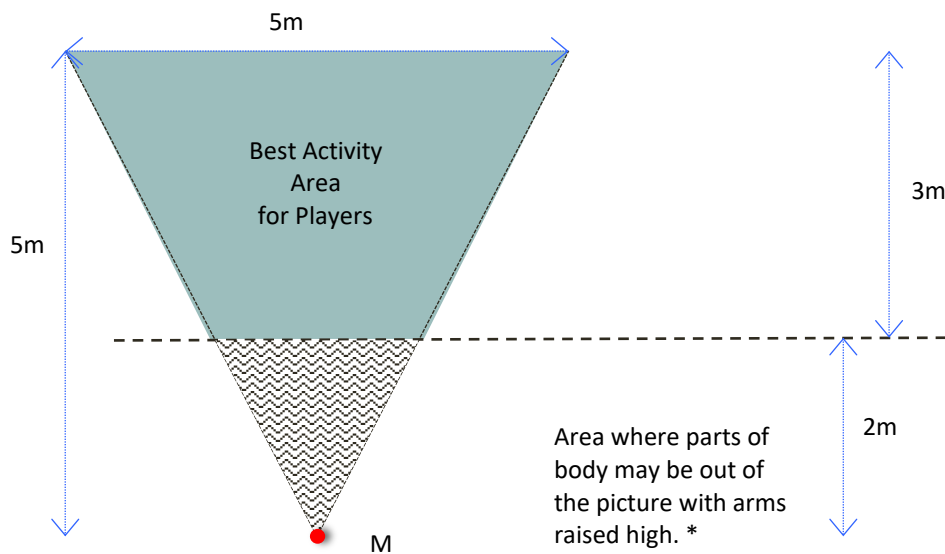
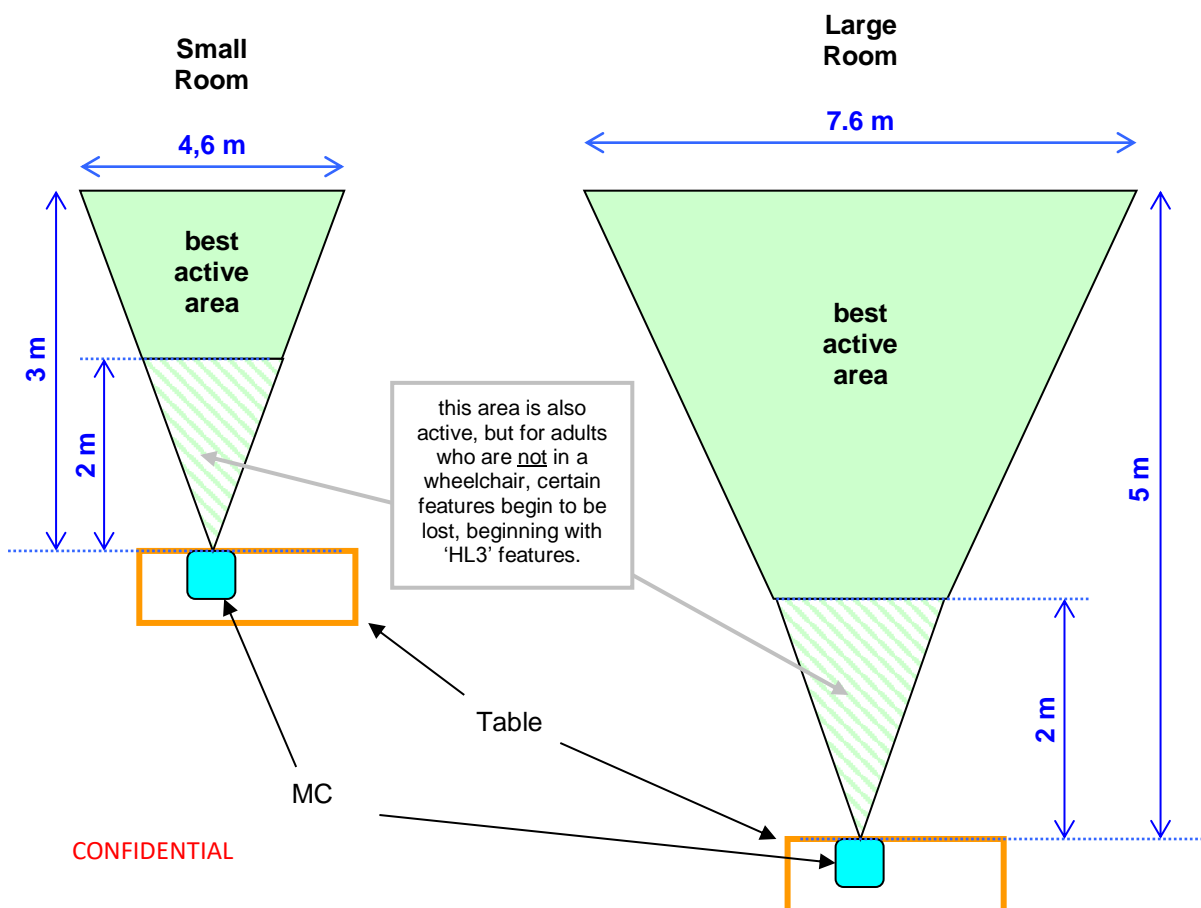


Figure 3: field-of-view

* The “too-close” point depends on how tall the player is, but at a circa 2-meters range, and with arms extended overhead, a typical person might be completely in view. When the player is too close, the MC should still operate, but the user receives a warning light, either in the GUI, and/or via the box’s indicator lights. The warning light essentially means, “All features may not work properly, starting with Arms Overhead...”

Small and Large Room Use

Note: We are examining having 2 Field-of-View options for the user, right from the beginning: I.e.



Advantages: users know how much space they will need for which features. Small room is great for many things, in fact, probably most users will prefer it. CenterX features are lost, and maybe 2 person modes (though this is not clear). For workshops, performances, or sports class, obviously the Large Room mode is used, but this is probably rarer.

Disadvantages: it's a new button! and a choice that must be made. Automating it is another option of course. I.e. you turn on the machine and it says "You are in Small Room Mode"... "Clear out more space, or move into a larger room for Large Room features".

See also the white paper: "MC – Size of Active Area", where tests with various lens were done.

Software Requirements

Software Structure

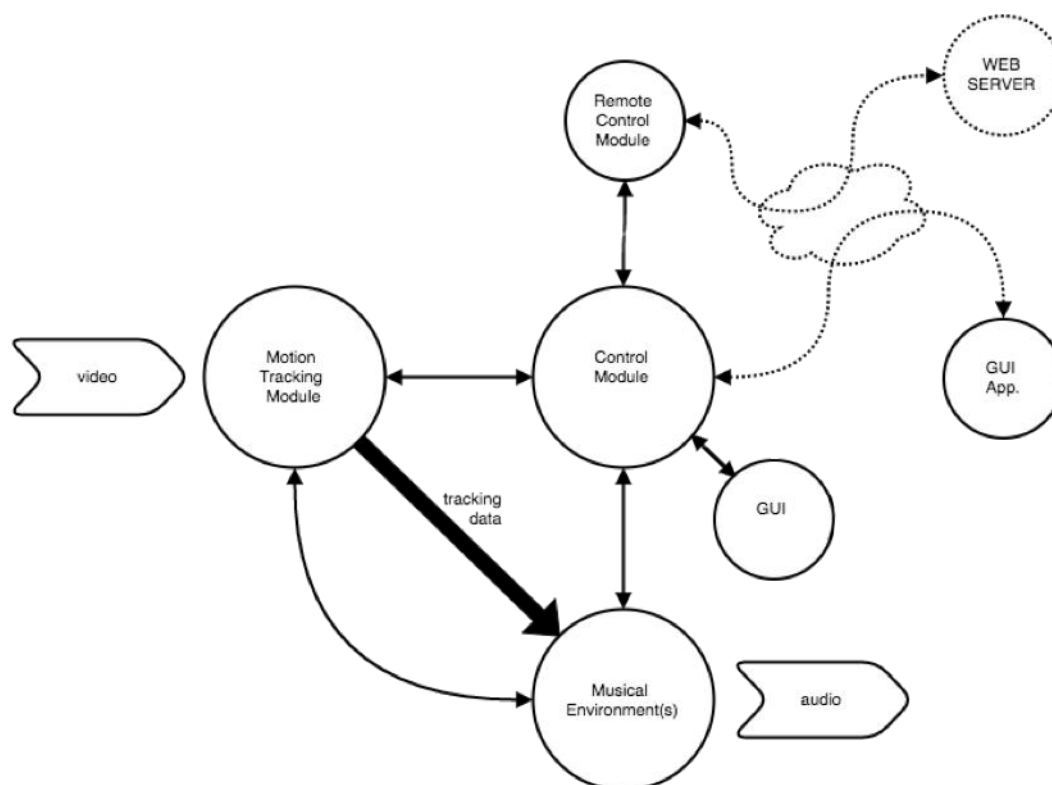


Figure 4: Software structure

Motion Tracking Module

TM is responsible for analyzing the video signals, recognizing the elements of motion alphabet and sending movement data as OSC to other modules.

General requirements

- Streams the video image to multiple GUIs through TCP Server
- Communicate with CM and MEs through OSC messaging
- The TM must be able to find the blobs -human forms- and send blob data to CM (Either standing, sitting in a wheelchair, or crawling or even if the player(s) are not moving)
- identify active players

- provide all of the OSC data streams referred to in the Movement Alphabet
- continuously track active players and send movement data to ME

Questions:

1. The frequency of video image update
2. The frequency of blob data update
3. What happens to the player assignment when a blob is lost? --- We need the work on irregular cases after implementing the regular behaviours.

Motion Tracking Schemes

The MC uses two schemes for tracking the player(s):

Player-Based

TM tracks the human form. It works for 1 or 2-players according to a selection made by the user via CM(GUI). 1-player mode is the default.

TM can recognize the human form standing, walking, sitting in a wheel chair or crawling on the floor and send blobs to the CM(GUI). The software identifies one or two bodies as the active player(s). Once this happens, the TM tries to stay with its selection even when other bodies enter the stage.

In this scheme the TM must be able to:

- recognize and mark as “active” the human form(s)
- track one or two active players and provide data on each, identified by user id
- recognize all movements and gestures in the Movement Alphabet (see MA) and send data accordingly
- allow selection/de-selection of active player (via GUI), as explained below:

Zone-Based

TM tracks activity in defined areas

Unlike Player-based motion tracking, Zone-based makes no attempt to find blobs. Thus, any number of people can use this scheme at the same time. They might be in a bed, on a mat on the floor or walking around.

Obviously, if a therapist walks into the environment, they too will be heard (and this is desired). It is thus simpler and more robust than Player-based tracking. Only two types of data are sent for each zone, a continuous activity level, and Discretes. See Movement Alphabet for details. In the GUI, the operator can see the screen divided into left and right zones. The line that divides the screen can be shifted left or right by the GUI interface.

In this scheme, the TM

- should receive zone vertex and id sent by GUI
- track zone(s) accordingly and provide data identified by zone id
- send normal and discrete activity data as described in the MA
- accept other control parameters sent by GUI
- permit zone size changes, sent by GUI, as described below:

Motion Tracking Modes:

According to the above mentioned scheme there are 3 modes

- 1 player mode
- 2 players mode
- Zone-only mode

Player Activation/Deactivation Scenarios:

Scenario 1

In this scenario, user clicks and select players intuitively without considering the modes.

In this scenario only message is **/set/player/[id]/tracking 1/0**

TM can automatically assign blobs to players, see the notes below the frames.

Only **/set/player/swap** message may be needed additionally to easily swap the IDs

Scenario 2 and 3

In these scenarios with screen interaction CM or TM needs to know which blob is clicked.

To handle this either

- the color-based-blob-selection proposal as Tobias made, may be a solution but I am not sure if it works on different screens and light conditions etc. Because the blob will be transparent so that the people behind will be visible. Therefore the blob color cannot be fixed
- or CM sends the location clicked on the screen to TM and TM find the blob clicked
- or you do not send the image but the coordinates of the blob every time the video image is refreshed and CM draws the blob over video image. So CM always knows about the blobs.

CM sends to tracking enabling data to TM only, not to MEs

```
/set/player/[id]/tracking ,i 1/0
```

```
/set/zone/[id]/ tracking ,i 1/0
```

MEs only react to the tracking messages received regardless of the Tracking Mode

Control Module

CM is the main loop of the MC3.0. It is responsible for

- core operations and communication among the modules
- attaching/detaching Musical Environments
- adding custom MEs to the system
- providing APIs for ME development
- communicating with GUI
- web connection for remote TM, CM, GUI, ME. software updates

For full functionality of CM please see the TM, GUI and MEs.

CM is the main module which starts when the device is turned on. CM starts other modules. All GUI functionality is in the CM but GUI is only the interface that transfers user actions to CM.

Graphical User Interface (GIUs)

All the functionality of the GUI is actually provided by the CM.

GIUs only send messages to and receives messages from CM

Only exception is TCP client which communicates directly with TM TCP Server to stream video image.

There are –at least- two GIUs:

1. mainGUI runs on the MC hardware and is accessible through an external monitor
2. GUIapp is available for users on a tablet which comes with the MC as part of the product. Alternatively, downloadable mobile device applications may be implemented later for multiple user control, etc.

General Features:

- installs an TCP client to receive video image from TM
- provide the user interface in 2 sections
 - initial GUI interface
 - the video image(from TM) with interaction capacities
 - ME selection tool
 - Link to Help, About ME, Feedback interface features
 - ME control interface
 - Common tools
 - Musical tools
 - Tracking tools

and GUI is used to

- check the video image, for example to be sure the player is the picture
- select the ME
- In Player-Tracking mode select/deselect players
- In Zone-Tracking mode, adjust the zones visually
- adjust the volume
- adjust the sensitivity – how much movement is required to achieve results
- change sound bank, add accent choices when ME provides the option
- restart ME

- start/stop ME

GUI Interface Elements:

Initial Screen: When GUI started (after initialization with TM)

- Video image with blobs over the video when provided by TM
- See following Player Selection details
- ME selection button (dropdown or a button for each ME)
- Help button
- Feedback button
- About button

ME Screen: When any ME activated

- Common controls
 - Volume
 - Sensitivity
 -
- Musical controls
 - Different for each ME (see ME section)
- Tracking Controls
 - 1 Player
 - 2 Players
 - Zones Only

Help Screen: When selected

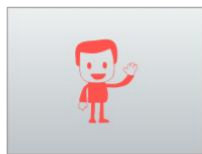
Feedback Screen: When selected

About Screen: When selected

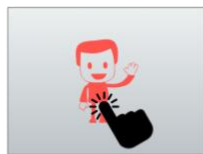
Player Selection

Scenario 1

This scenario is actually based on no “1 player / 2 players” selection buttons installed.



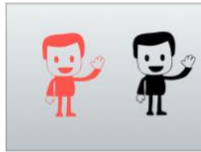
Player enters
> selected auto
> id = 0
> OSC
* if two enter one is selected



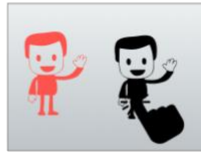
Clicked on...



Player deselected
> No OSC



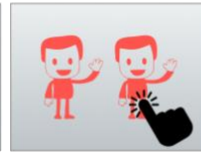
New Player enters
> no change
selected one continues
> id = 0
> OSC



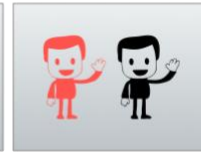
Clicked on...



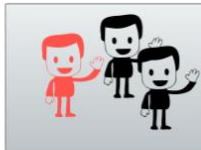
New Player selected
> id = 2
> OSC
Initially selected has a new id
> id = 1
> OSC



Clicked on...



New Player deselected
Initially selected has a new id
> id = 0
> OSC



3rd Player enters
> no change
selected one continues
> id = 0
> OSC



Any player (2nd in this example) is out of view then she is on-hold
> id = 2
> no OSC
Initially selected has a new id
> id = 1
> OSC



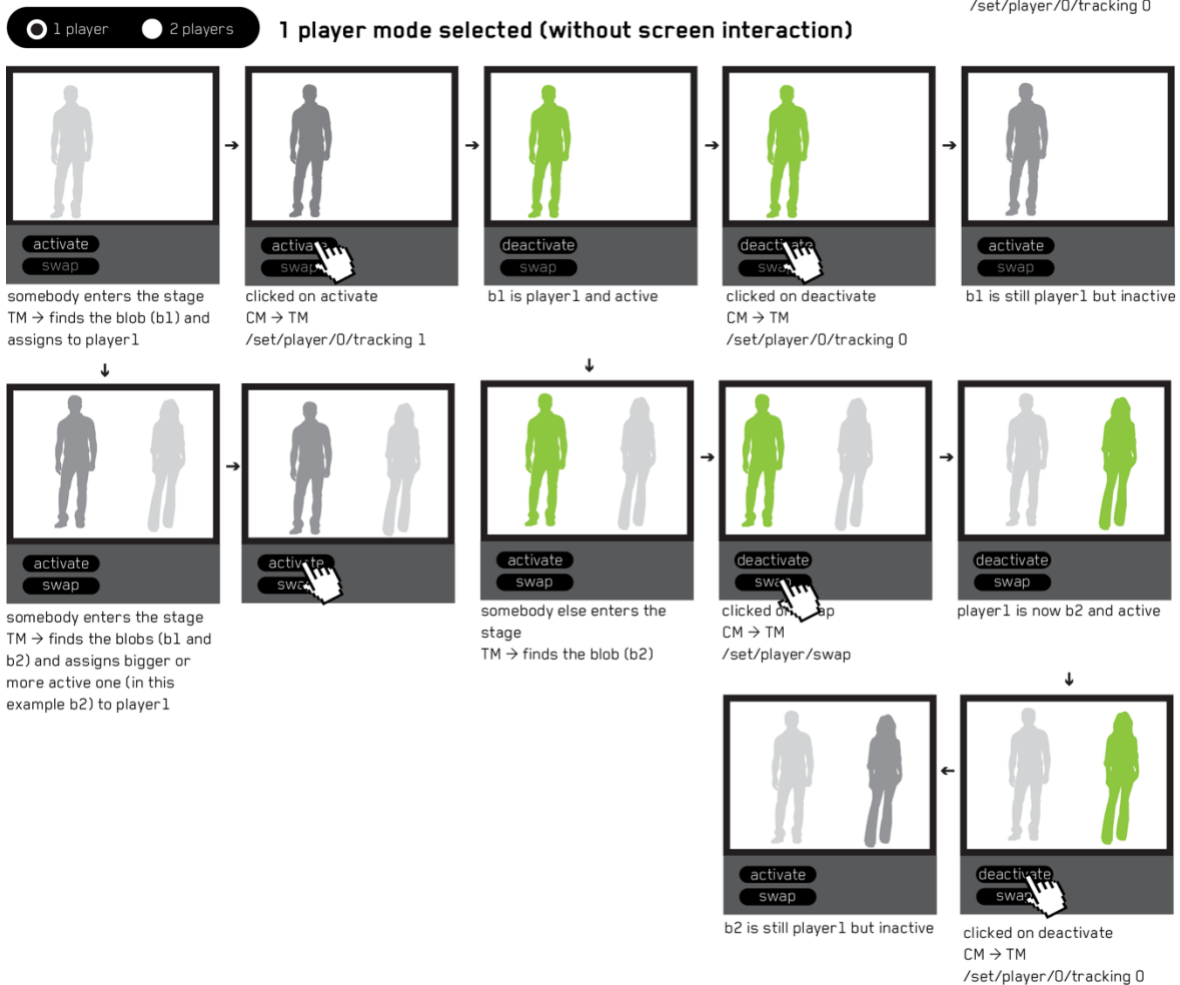
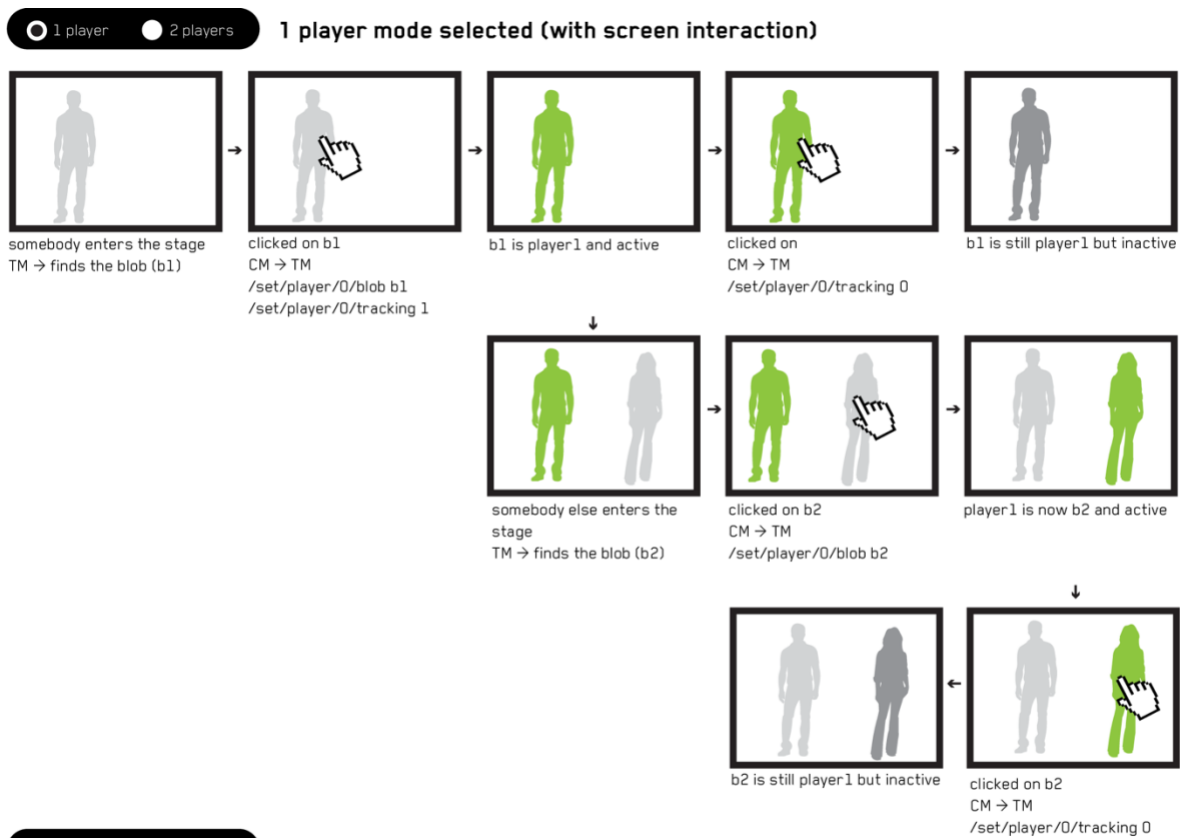
When she or anyone back in the view on-hold becomes selected
> id = 2
> OSC
Initially selected has a new id
> id = 1
> OSC



3rd Player enters
> no change
selected one continues
> id = 0
> OSC

Scenario 2

This scenario is based on existence of "1 player / 2 players" selection buttons installed and "1 player" is selected. The first set of drawings show the screen interaction over the video, this is the main interaction style. The second set of drawing shows the interaction method without screen (video) interaction but additional buttons. This is added to be ready to implement in case the user has difficulties using the screen interaction over video.



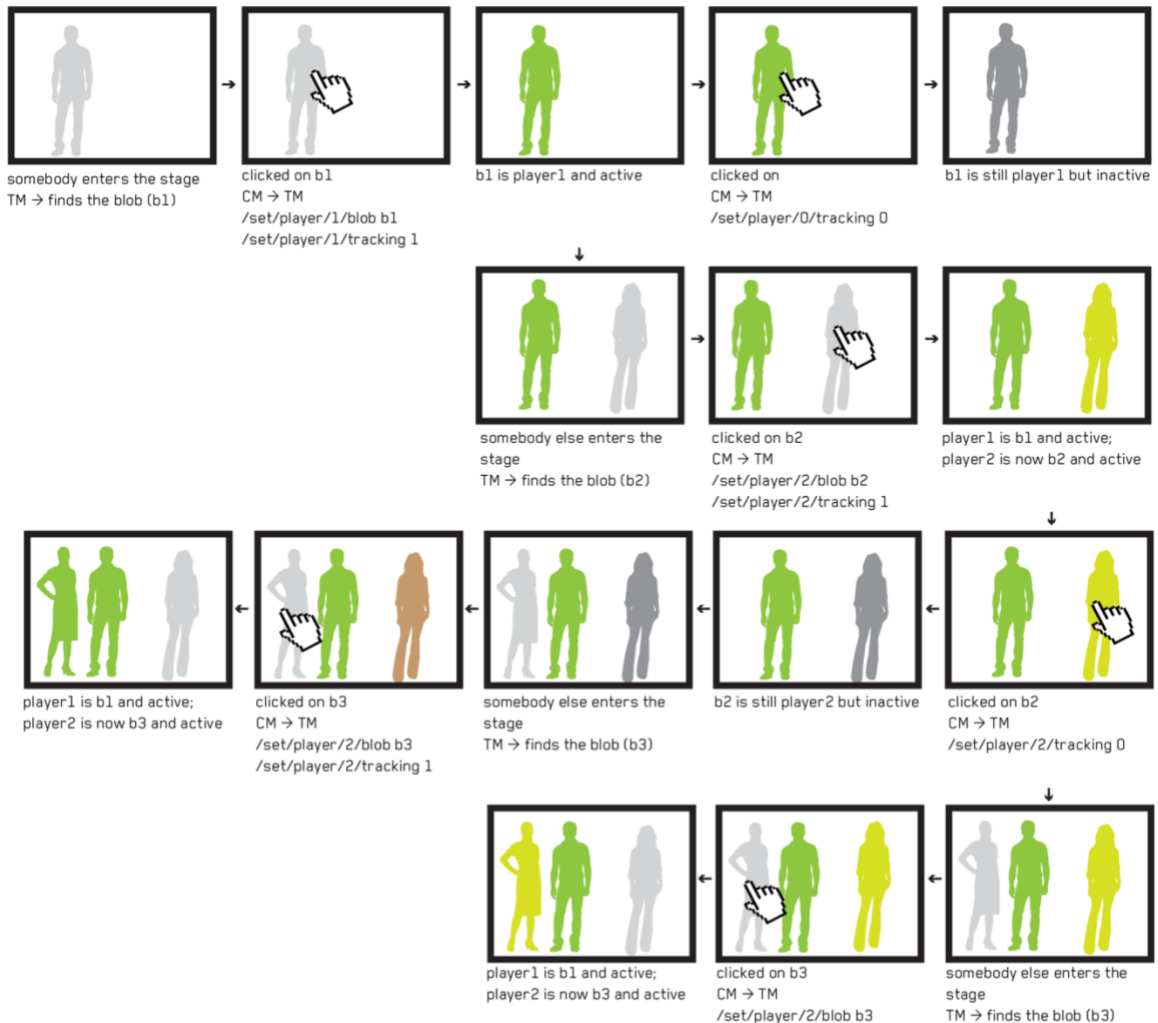
Scenario 3

This scenario is based on existence of “1 player / 2 players” selection buttons installed and “2 players” is selected.

The first set of drawings show the screen interaction over the video, this is the main interaction style. The second set of drawing shows the interaction method without screen (video) interaction but additional buttons is added to be ready to implement in case the user has difficulties using the screen interaction over video.

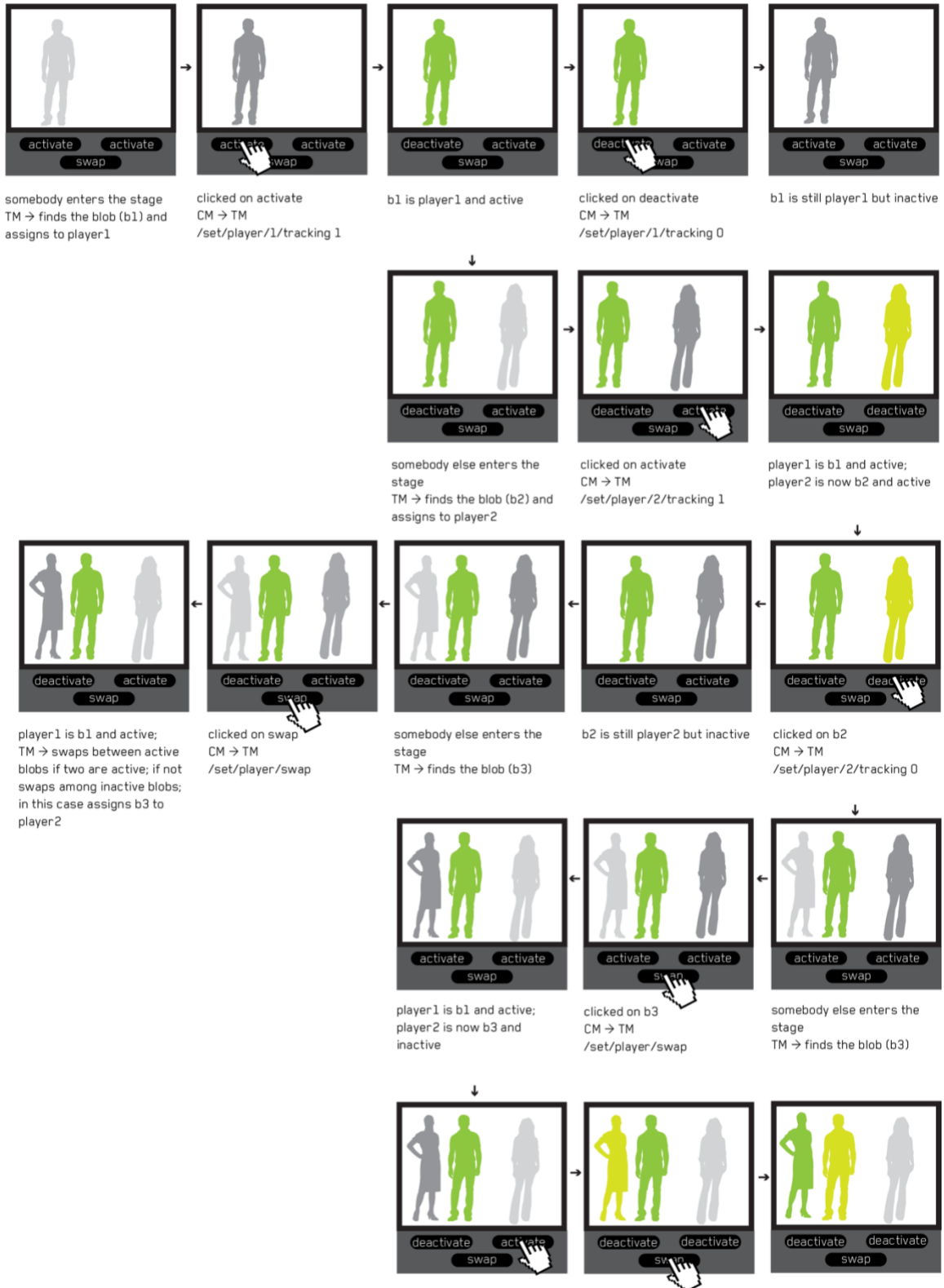
☒ 1 player ☐ 2 players

2 players mode selected



1 player 2 players

2 players mode selected (without screen interaction)



Messaging Protocols Among Modules

Message Addressing Protocol

Modules have their port id's

CM : 65

TM : 61

ME(s) : 60

GUI : 62

OSC client (sender) and server (receiver) ports are assigned alike: sender port id + receiver port id

So CM send messages to TM on port 6561 and listens TM on 6165

And TM listens CM on port 6561 but sends to CM on 6165

So complete list is:

CM sends to TM on 6561

CM sends to ME on 6560

CM sends to GUI on 6562

CM listens TM on 6165

CM listens ME on 6065

CM listens GUI on 6265

TM sends to CM on 6165

TM sends to ME on 6160

TM listens CM on 6561

TM listens ME on 6061

ME sends to TM on 6061

ME sends to CM on 6065

ME listens TM on 6160

ME listens CM on 6560

This arrangement is needed in order to prevent trying to bind the same port which gives an error since it is not implemented in all programming environments used.

Conventions

In all messages that include players' id,

like /set/player/[id]/<anything follows>

ex: "/set/player/0/soundbanks/[#]/list piano flute guitar"

always

[id] = 0 indicates 1st player

[id] = 1 indicates 1st player

[id] = 2 indicates 2nd player

ex: "/set/player/0/soundbanks/[#]/list piano flute guitar" sounbank list for 1st player

ex: "/set/player/1/soundbanks/[#]/list piano flute guitar" sounbank list for 1st player

ex: "/set/player/2/soundbanks/[#]/list piano flute guitar" sounbank list for 2nd player

BUT

For the zones 0 and 1 respectively indicates zone 1 and zone 1

ex: "/set/zone/0/soundbanks/[#]/list piano flute guitar" sounbank list for 1st zone (= zone 1)

ex: "/set/zone/1/soundbanks/[#]/list piano flute guitar" sounbank list for 2nd zone (= zone 2)

AND

No [id] > indicates both players, like

/set/player/<anything follows>

ex: "/set/player/soundbanks/[#]/list piano flute guitar" sounbank list for player 1 & player 2

and both zones, like

/set/zone/<anything follows>

ex: "/set/zone/soundbanks/[#]/list piano flute guitar" sounbank list for zone 1 & player 2

Musical Environment(s) Initialization Sequence

CM starts the selected ME application. Then

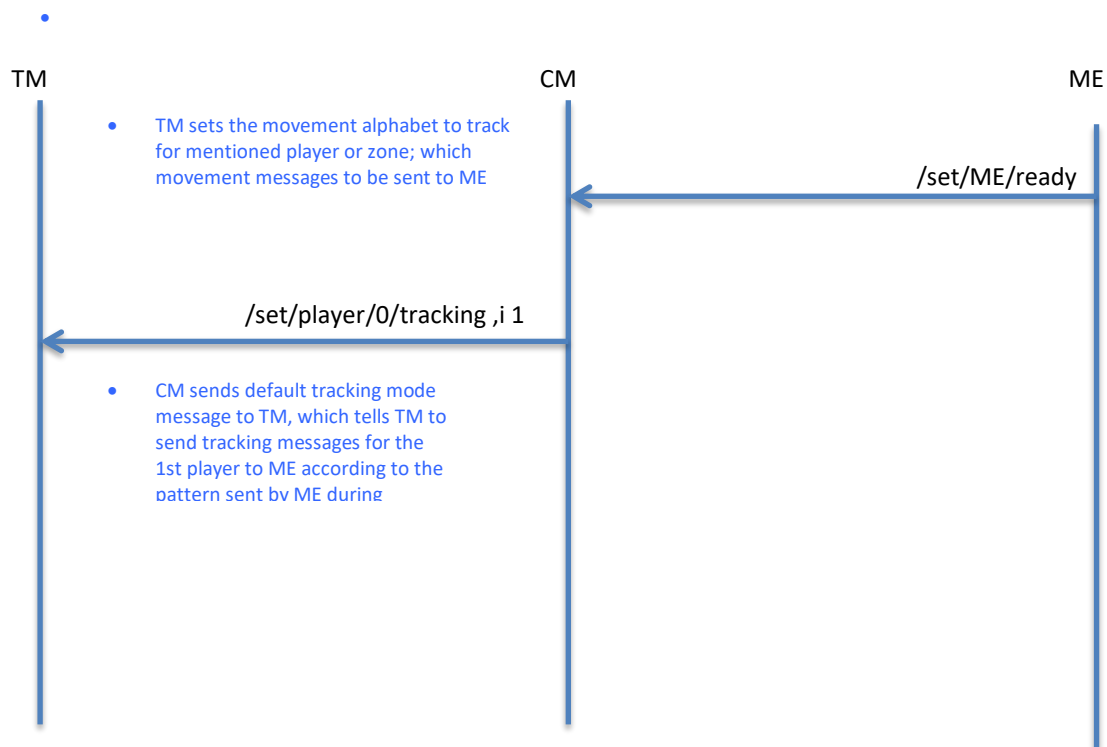
#	direction	OSC message address pattern, typetag, argument			information
1	ME to CM	/set/ME/loaded	None	None	On load; successful start of application
2	CM to ME	/set/initialize	None	None	Request initialization (Else then loading process, this message can be sent any time and starts the following messaging sequence)
3x	ME to CM	/set/player/[id]/soundbank/[#]/list and/or /set/zone/[id]/soundbank/[#]/list	,sss	names of the sounds	to set the GUI elements; as many as needed
4x	ME to TM	/set/alphabet/[pattern] examples: (There is a trick here, which is sending the message with or without [id]. without [id] means message is for all players or for all zones) /set/alphabet/player/activity/discrete 1 (for both player) /set/alphabet/zone/activity/discrete 1 (for all zones) /set/alphabet/player/[id]/activity/discrete 1 /set/alphabet/player/[id] 0 /set/alphabet/player/[id]/activity 1 /set/alphabet/zone/[id]/activity 1 etc...	,i	1/0	To request needed tracking messages only; as many as needed. (This message can be sent anytime when needed to change the tracking scheme)
5	ME to CM	/set/ME/ready	None	None	Initialization completed; ready to operate
-	ME to CM	/set/error	,i	Error number	In case. See error message for error types.

Tracking Module Initialization Sequence

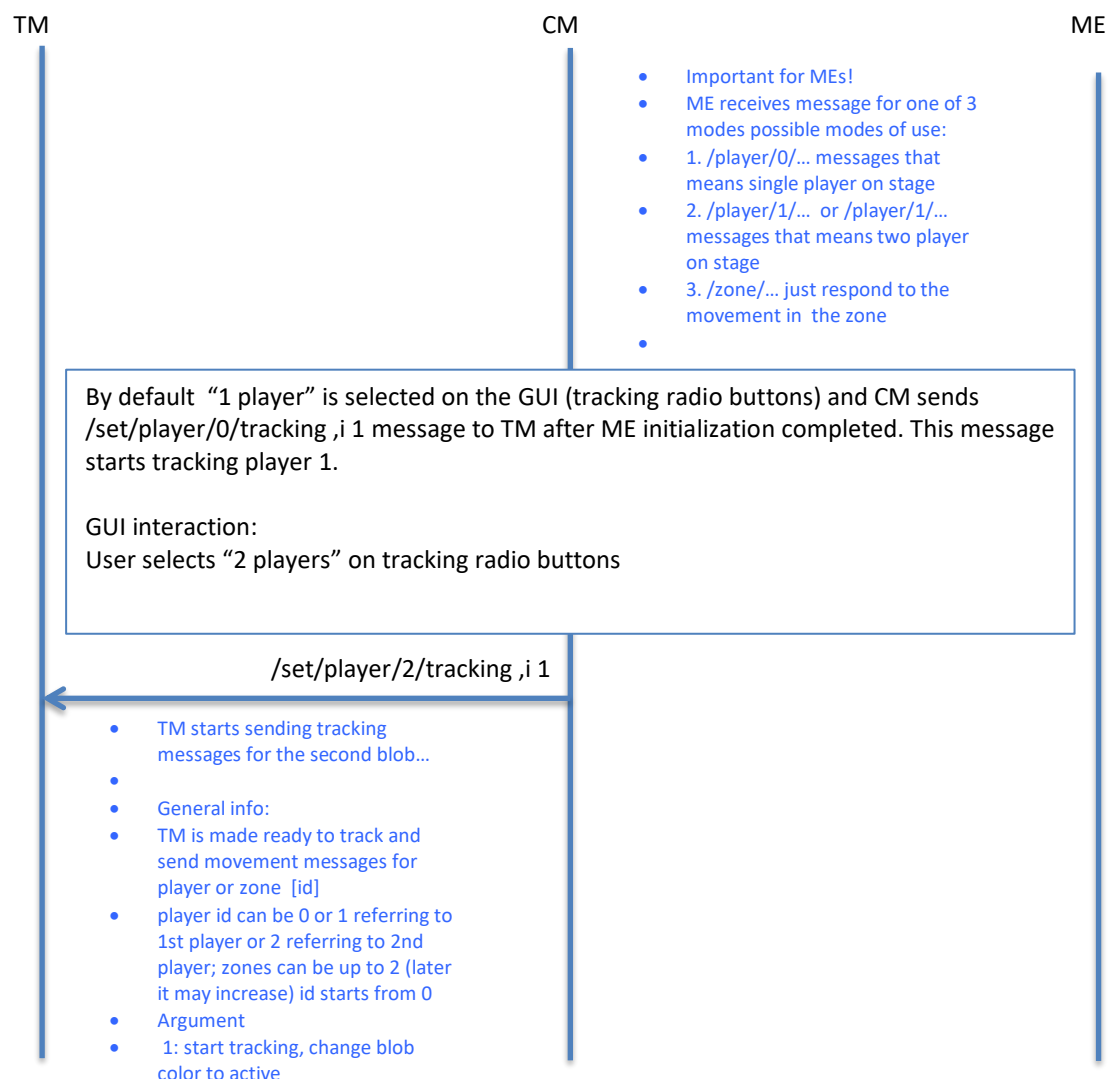
#	direction	OSC message address pattern, typetag, argument			information
1	TM to CM	/set/TM/loaded	None	None	On load; successful start of application
2	CM to TM	/set/initialize	None	None	Request initialization (Else then loading process, this message can be sent any time and starts the following messaging sequence) On initialize message TM <ul style="list-style-type: none"> retrieves the video image, detects players and starts drawing blobs for one or more people in inactive color and saves image on hard drive in continuous manner to be used by CM <ul style="list-style-type: none"> the frequency of updating image to be decided the size of the image to be decided (400x300px I suggest)
3	TM to CM	/set/TM/ready	None	None	That means initialization is completed and video image frames started streaming
4x	CM to TM	/set/player/[id]/tracking	,i	1/0	Start tracking person [id], convert blob color to active and start sending messages. * here an option may be, not sending movement messages before receiving ready message from ME
5	CM to TM	/set/zone/[id]/blob	,iiii	x1,y1,x2,y2	Register the zone
5	CM to TM	/set/zone/[id]/tracking	,i	1/0	Start tracking zone [id], sending messages. * here an option may be, not sending movement messages before receiving ready message from ME
-	TM to CM	/set/error	,i	Error number	In case. See error message for error types.
4x	ME to TM	/set/alphabet/[pattern] examples*: /set/alphabet/player/[id]/activity/discrete /set/alphabet/player/[id] /set/alphabet/player/[id]/activity *: see OSC messages section for more	,i	1/0	To request needed tracking messages only; as many as needed. (This message can be sent anytime when needed to change the tracking scheme)
	CM to TM	/set/tracking	,i	1/0	Start or stop sending tracking messages.

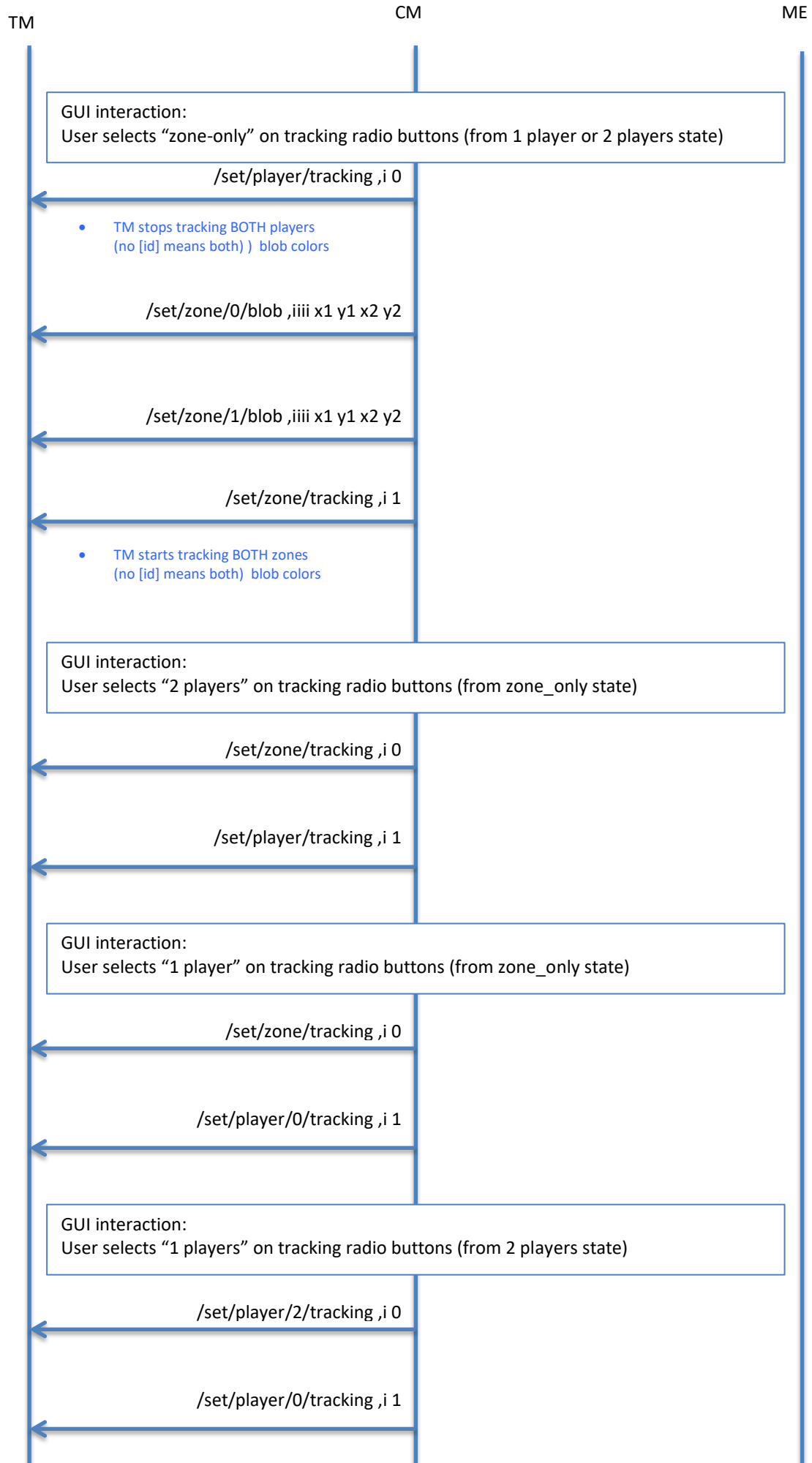
CONFIDENTIAL

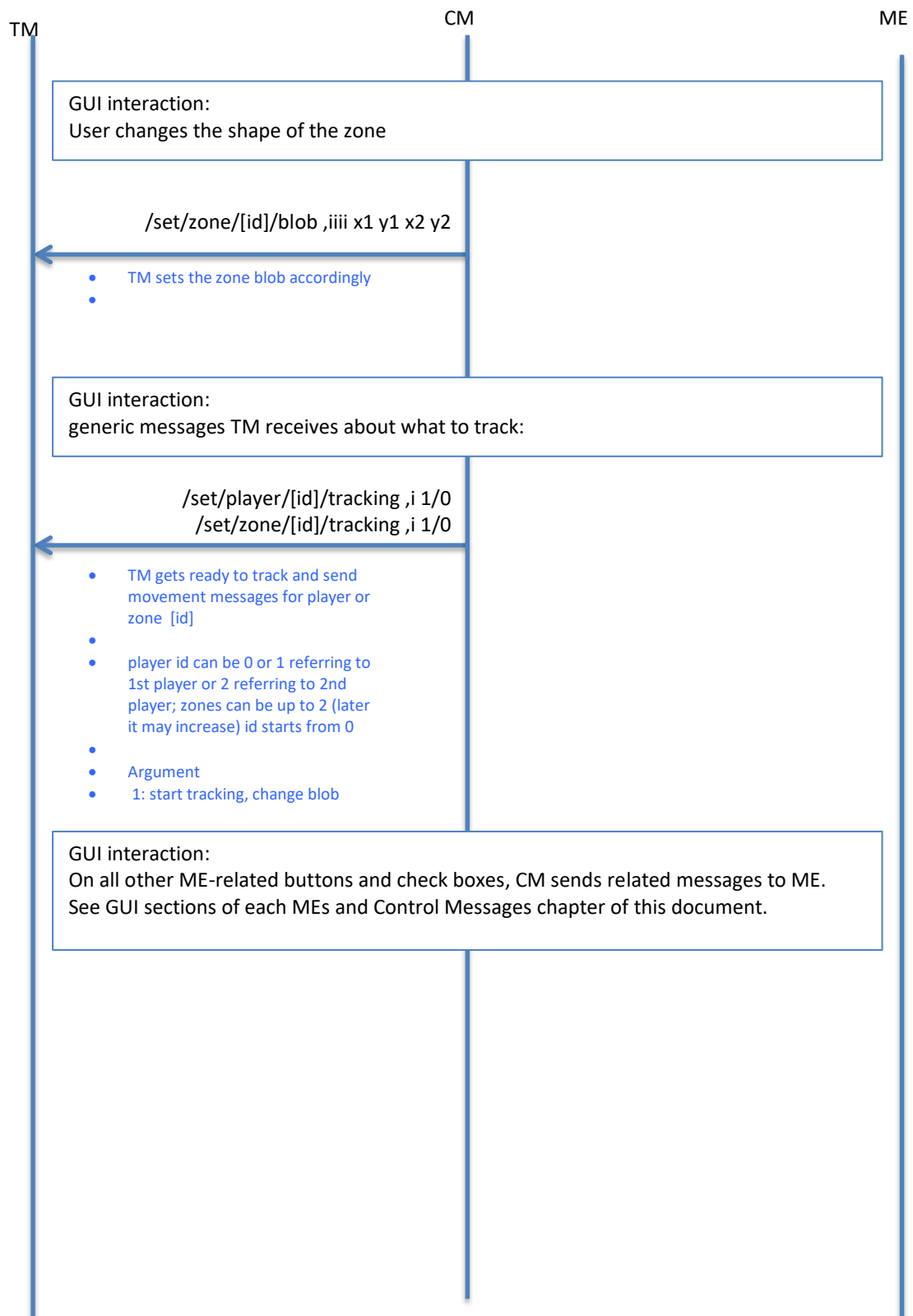




USER CONTROL MESSAGING SEQUENCES







Control Messages

Message Number	Source/Destination to/from	OSC Messages						When	Function*
**		Address pattern				type tag	Arguments		
S01	MEtoMT	/set	/alphabet	/[pattern]*		,i	1/0	anytime	Sets list of messages from the alphabet to be tracked or not
S02	CMtoMT		/player/[id] or /zone/[id]	/tracking		,i	1/0	anytime	Start tracking player or zone on/off
S03	CMtoMT		/player or /zone	/swap		NA		On operator request	Swap id's of player or zone
S04	CMtoME		/initialize			NA		anytime	Initializes ME
S05	CMtoME		/volume			f	normalized volume value	On operator request	adjust volume
S06	MEtoCM		/player/[id] or /zone/[id]	/soundbank/[#]	/list	s (as many)	Names of the sounds	At initialization	Sends the list of soundbanks
S07	CMtoME		/player/[id] or /zone/[id]	/soundbank/[#]	/sound	i		On operator request	Selects a soundbank
S08	CMtoME		/evolving			,i		On operator request	Set evolving on/off
S09	CMtoME		/tracking			,i	1/0	On operator request	Start/Stop all tracking
S10	MEtoCM		/ME/loaded			None		When loaded	ME loaded
S11	CMtoME		/play			,i	1/0	On operator request	Start / Stops ME (playing)
S12	CMtoME		/sensitivity			f		On operator request	Sets sensitivity
S13	TMtoCM		/TM/loaded			None		When loaded	TM loaded
S14	CMtoTM		/zone/[id]/	/order	/up /down	NA		On operator request	Readjust the zone id's
S15	CMtoTM		/zone/[id]/	/blob		,iiii	x1,y1,x2,y2		Sets blob coordinates
S16	MEtoCM		/ME/ready			NA		On successful initialization	ME initialized
S17	TMtoCM		/TM/ready			NA		On successful initialization	TM initialized
S18			/player/[id] or /zone/[id]	/mute		,i	1/0	On user interaction	Mutes/activates the sound of player or zone
S19			/player/[id]	/inPlace		,i	1/0	On user interaction	Sets in-place (vertical) playing
S20	ANYtoCM		/error			s	ERROR NUMBER	anytime	On error
S21	TMtoCM		/blob/[id]			,iiii	x1,y1,x2,y2,...	continuously	GUI draws the blobs
S22	CMtoTM		/player/[id]	/blob		i	Blob id	On user interaction	

G01	CMtoME	/get	/player/[id] or /zone/[id]	/soundbank/[#]	/list	NA		anytime	Requests soundbank list
G02	CMtoMT		/player/[id] or /zone/[id]	/blob		NA			Requests the blobs
G03									
G04									

GUI APP is the application runs on iOS and Android devices that communicates over wifi and provides additional GUI.

MOVEMENT ALPHABET

Introduction

The Movement Alphabet is a list of body-to-sound mappings meaningful in the context of MC. Some are Player-based -- based on the body(ies) of 1 or 2 player(s), and some are Zone-based -- based on action in the room, without regard to the human form.

Player-based:

- Activity - how much and in what direction (flow) we are moving
- Location - where we are in the room
- Position - shape of the body and orientation of its parts (also referred to as "form")
- Gesture - a short movement with communicative intent

Zone-based:

Zones (zone-based) – delivers only two types of data

Within each type of movement data, there is a hierarchy of sub-types, sub-sub-types, etc.

Note:

virtually every human movement can be interpreted in multiple ways. If you move your left hand quickly to the left, for example, this action could be represented in the following ways:

Description in words	OSC data name
activity occurred	/activity
your left hand was still, followed by movement.	/activity/discrete/hand/left
there is a continuously changing value, according the amount of movement in that area	/activity/normal/hand/left
there is a continuously changing value, according the amount of movement in the upper body	/activity/normal/body/upper
the arm is now extended to the side	/position/side/hand/left
if it is a quick movement to the side, then it is also interpreted as a gesture called, "hit"	/gesture/hit/side/left
starting from stillness you move, but in this case it is function of the place you are in the room, not the part of your body involved.	/zone/activity/discrete
etc. there are others	etc.

When an ME is launched, it calls for only certain Messages to be sent. Thus, while the list below shows all of the possible data streams, only a few are available at any one time, depending on the ME that is running.

The Alphabet Table

Message Number	OSC Messages							When	Priority*
**	Tracking paradigm	Id	Movement Type	Feature	Body Part	Attribute	Data		
T01	/player	/[id]	/activity	/discrete	/hand	/left	NA	On occurrence	1
T02						/right	NA	On occurrence	1
T03					/head		NA	On occurrence	1
T04 *					/leg	/left	NA	On occurrence	1
T05 *						/right	NA	On occurrence	1
T06					/body	/upper	NA	On occurrence	1
T07						/lower	NA	On occurrence	1
T08						/left	NA	On occurrence	1
T09						/right	NA	On occurrence	1
T10				/normal	/hand	/left	,f	Continuous	1
T11						/right	,f	Continuous	1
T12					/head		,f	Continuous	1
T13 *					/leg	/left	,f	Continuous	1
T14 *						/right	,f	Continuous	1
T15					/body	/upper	,f	Continuous	1
T16						/lower	,f	Continuous	1
T17						/right	,f	Continuous	1
T18						/left	,f	Continuous	1
T19				/peak			NA	On occurrence	1
T20				/flow	/leftwards	/left	NA	On occurrence	2
T21						/right	NA	On occurrence	2
T22					/rightwards	/left	NA	On occurrence	2
T23						/right	NA	On occurrence	2
T24					/upwards	/left	NA	On occurrence	2
T25						/right	NA	On occurrence	2
T26					/downwards	/left	NA	On occurrence	2
T27						/right	NA	On occurrence	2
T28	/player	/[id]	/location	/ready			NA	On occurrence	1
T29				/present			NA	On occurrence	1
T30				/centerX			,f	Continuous	1
T31				/centerZ			,f	Continuous	3
T32				/outOfRange			,s	On occurrence	2
T33	/player	/[id]	/position	/height			,f	Continuous	1
T34				/heightLevel			,i	On change	1
T35				/vertical	/hand	/left	,f	Continuous	1
T36						/right	,f		1
T37				/side	/hand	/left	,f		1
T38						/right	,f		1
T39					/foot	/left	,f		2 n
T40						/right	,f		2n
T41				/front	/hand	/left	,f		1
T42						/right	,f		1
T43					/foot	/left	,f		2n
T44						/right	,f		2n

T45				/width			,f		1
T46	/player	/[id]	/gesture	/hit	/overhead		NA	On occurrence	1
T47					/side	/left	NA	On occurrence	1
T48						/right	NA	On occurrence	1
T49					/down	/left	NA	On occurrence	2/3
T50						/right	NA	On occurrence	2/3
T51					/forward	/left	NA	On occurrence	1
T52						/right	NA	On occurrence	1
T53				/kick	/side	/left	NA	On occurrence	1
T54						/right	NA	On occurrence	1
T55					/forward	/left	NA	On occurrence	1
T56						/right	NA	On occurrence	1
T57				/doubleArmSide			NA	On occurrence	1
T58				/doubleArmSideClose			NA	On occurrence	1
T59				/jump			NA	On occurrence	2
T60				/clap			NA	On occurrence	3
T61	/zones	/[id]	/activity	/normal			,f	Continuous	1
T62				/discrete			NA	On occurrence	1
T63				/flow	/leftwards				2
T64					/rightwards				2
T65					/upwards				2
T66					/downwards				2

Figure 10: Tracking messages as sent from TM to MEs

* priorities:

1 = needed

2 = desired... there are no ready uses, but we believe that it could be useful.

2n = there's no known use for this

3 = interesting... but we fear may be difficult to program.

** tracking messages coded with 'T', go from TM to ME

* - note: I do not see why we need this. it is equal to "lower body", etc. There is no "leg recognition" per se. Not only is it technically difficult, but I cannot think of any reason for it. (I cannot remember why we might have added it) ??? The point is while we want to be thorough, we do not want to add features which we do not need (if they take a lot of work to do!).

MC3 is based on the features we need. We can expand it later.

ee>>>: as we see in the latest tracking algorithm Andreas sent these things are becoming easier every day. But also for several other reasons the movement of the leg can be important –also for gestures- and different than lower body. Let us keep them.

Alphabet Features

Activity (T01-T27)

Discretes (T01-T09):

Discrete activity is any movement starting from stillness. These can be short actions, like a twitch, or blink of an eye, But note, any movement following a stillness is interpreted as a Discrete.

Discrete Hand (T01,T02):

Discrete movement of hands; Left and Right differentiated.

CONFIDENTIAL

Discrete Head (T03):
Discrete movement of head.

Discrete Leg (T04,T05): * - see above.
Discrete movement of legs or feet; Left and Right differentiated.

Discrete Body (T06-T09):
Discrete movement of upper body, lower body, left half of body, right half of body.

Normals (T10-T18):

These are typical human movements, and there is a large dynamic range. The MC2.0 does not have the ability to track activity of a hand or other body part, so it is not a high priority for the MC3.0 either, but having it would allow us to prioritize, or focus on certain body parts and not others. A patient may, for example, have controlled movement in one body part, and uncontrolled movement in another, so it would be important for her or him to hear the part they can control and ignore other parts.

Normal Hand (T10,T11):
Continuous movement of hands; Left and Right differentiated.

Normal Head (T12):
Normal movement of head.

Normal Leg (T13,T14): * - See above.
Normal movement of left legs; Left and Right differentiated.

Normal Body (T15-T18):
Normal movement of upper body, lower body, left half of body or right half of body.

Peak (T19):
This refers to “peak energy”, i.e. the largest movement a player can make. It can be used to give an acoustic “prize” when it is reached.

Flow (T20,T27):
While it is not used in the MC2.0, information concerning the direction of movement could be useful. (I have heard direction information referred to as “flow”). Distinguishing left-flow and right-flow, for example, could be helpful in therapy. It also might help solve some of the Activity Problems discussed below in the Special Issues section.

SPECIAL ISSUES – 1 Problems with Activity Data

One of the biggest challenges of our project, is to deliver a convincing sense of causality. Some of the gestures do this quite well – such as “hits” – but are only used in certain cases. Another is Activity/Discrete, but for these to be experienced, the user must first hold perfectly still, and many people never do this.

The best all-round solution to causality is Activity/Normal. It offers a strong sense of expression and it universal – it belongs to every musical environment; and yet it has challenges of its own:

- Small gentle movements can produce erratic data, with values jumping up and down. This means that the quality of the resulting music will not match the quality of the movement (note: normal methods of filtering cause unacceptable latency). We wonder if Flow may help with this.
- Large gentle movements of the whole body produce much higher data values than energetic smaller movements. Again, this leads to a quality mismatch. We wonder if Flow may help with this as well.
- When a player is making large movements and the data values are at maximum, this is a problem since it does not give the player any motivation to move even larger. (We suggest some kind of non-linear mapping, so that it becomes virtually impossible for someone to ever reach a 100% value.)

- When the player moves closer to the camera, her data values get higher and higher, even

Location (T28-T32)

Ready (T28):

When the player walks in to play an ME which is played standing or sitting in place, then we do not want to hear music until they have finished walking. For example, “when player is present, and when the value of their /centerX changes less than 10% over a 1 second period of time, then send /ready”. *Still being discussed.*

Present (T29):

When the player walks in the vision. *Still being discussed.*

Center-X (T30):

The midline of the body in X-axis (perpendicular to the camera axis); on plane x. (There is no Center Y. Height and HeightLevel is used instead.)

Center-Z (T31):

The front of the body in Z axis; movement towards and away from the camera (not a high priority).

Center-Z is also used for compensation; when the player moves close to the camera (e.g. 2 meters away), the data streams should have roughly the same values as when the player is far away (e.g. 5 meters away).

Out of Range (T32):

Detection if the player is either too close to the camera, or too far away.

Position (T33-T45)

Position refers to the shape or form that the body makes.

Height (T33):

The highest point on the body. (see Special Issues, below.)

SPECIAL ISSUES – 2

Problems with Height and HeightLevel

The MC2.0 has had a persistent problem with height, namely, that when someone first walks in, the height takes a moment to come up from zero. Since /height is usually dependent on /heightLevel, a solution is to make /heightLevel default = 2, instead of 0.

Also, when the player leaves, height drops quickly to zero, and so the ending of music is radically affected and sounds terrible. One possible solution is to say: 1) if no one is present, prevent /heightLevel/0, and instead send /heightLevel/2, and, 2) if /centerX is near maximum or near minimum, block HeightLevel0.

Height Level (T34):

Height-level concerns posture. It is divided into 4 levels; thus delivers an integer value of 0,1,2 or 3. (see Special Issues, below.)

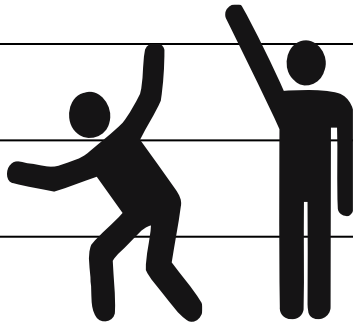
	/HeightLevel 3 reaching hands up overhead
	/HeightLevel 2 normal height movements
	/HeightLevel 1 crouching low i.e. knees bent, bending forward.
	/HeightLevel 0 lying on the floor, or crawling on the floor.

Figure 11: Height-level

Vertical Hand (T35,T36):

Right hand's vertical position. Many persons with Cerebral Palsy ("Kinderlähmung") cannot extend their arms well, so we want this to also work for arms that are not well-extended, for example, using elbows. ; Left and Right determined



Figure 12: hand height

Side Hand (T37-T38):

The distance of the right hand from the side of the body; in plane x; Left and Right determined.

Side Foot (T39-T40):

The distance of the right foot from the side of the body; in plane x; Left and Right determined.

Front Hand (T31,T42):

The distance of the hand from the front of the body; in plane z; Left and Right determined.

Front Foot (T43,T44):

The distance of the foot from the front of the body; in plane z; Left and Right determined.

Width (T45):

Width is the distance between the left-most and right-most points on the body, i.e. the player's total width, changing continuously. Actually refers to the blobs maximum width.

Gesture (T46-T60)

Gesture refers to communicative physical actions. They are all Boolean in nature.

Hits (T46-T54):

Hits are sudden outward-from-the-body movements of hands, elbows or feet. There are different kinds of Hits, some of them represented by the colored blocks in the figure below and others are explained.

Hit Overhead: ■□

Hands go quickly overhead.

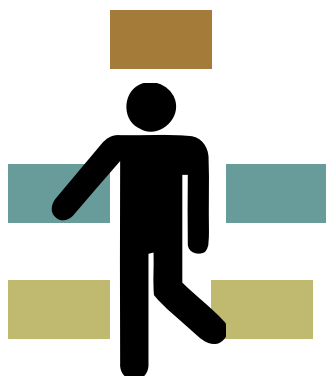


Figure 13: hits side, overhead, and kick side

Hit Side:

Hands go quickly out to the side.

Left and right hands are differentiated.

Hit Down:

Imagine you are playing a drum -- you hit your hand quickly downward in the air

Hit Forward:

Hand goes forward towards the camera. Left and right are differentiated.

Kick (T55-T56):

Foot goes sides or forward towards the camera. Ideally, Kick is a Priority 1 gesture, but the differentiation of Left and right is less important

Double Arm Side (T57):

Opening both arms quickly to the side. `doubleArmSide` is a toggle type of controller, i.e. the closing of the arms is also sometimes used to turn "off" the situation that `doubleArmSide` has turned "on". There is Delta on opening, but not closing. In other words, opening the arms slowly will not turn on `doubleArmSide`, but closing them slowing will turn it off.

Double Arm Side Close (T58):

Closing both arms quickly to the center, opposite of DAS.

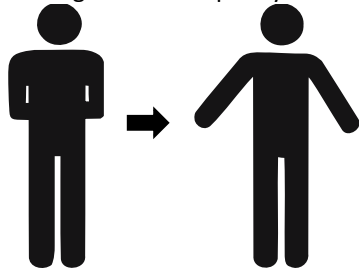


Figure 14: Double Arm Side

Jump (T59):

Both feet have to come off the floor by ca. 10 cm.

Clap (T60):

Bringing hands together. It does not need to be done quickly. The clap makes a useful switch-gesture, for example for the player to change the sound bank. This is a nice robust feature in the Kinect2 (where I got the idea). Of course, we realize this may not be so easy, that is why we listed it as a low priority.

Zones (T61,T66):

There are two zones in the MC3.0. (In later versions there can be more zones, see the discussion in Pro-Version). Zones offer two types of data:

Zone Discrete (T61):

Discrete movement in zone

Zone Normal (T62):

Normal continuous movement in the zone

Zone Flow (T63-T66):

Direction of the movement in the zone

OSC Message Reference

Open Sound Control (OSC) is a protocol for networking and controlling sound devices such as synthesizers and computers (<http://opensoundcontrol.org/>). The following section shows how messages are sent between modules in the MC.

OSC specification, terminology and MC conventions:

Client and Server

CONFIDENTIAL

Client : sends messages
Server: receives messages

Currently we have 3 modules ME (Music Environments - all), CM (Control Module) and TM (Tracking Module)
Each module has to create a server on a special port
ME server port 6001 (for all incoming messages)
CM server port 6501 (for all incoming messages)
TM server port 6101 (for all incoming messages)
Each module has to create two clients on corresponding ports to send messages to others

OSC Message Structure:

Address pattern – Typetag – Argument

Address Pattern: is the message as we know it i.e. “/centerX”

Typetag: is the data type of the argument sent;

MC use atomic data types only :

,f: 32-bit big-endian IEEE 754 floating point number

,i: 32-bit big-endian two's complement integer

,s: OSC-string A sequence of non-null ASCII characters followed by a null, followed by 0-3 additional null characters to make the total number of bits a multiple of 32. (OSC-string examples) In this document, example OSC-strings will be written without the null characters, surrounded by double quotes.

,b: OSC-blob An int32 size count, followed by that many 8-bit bytes of arbitrary binary data, followed by 0-3 additional zero bytes to make the total number of bits a multiple of 32.

Argument: is the value sent with address pattern. It can be null as well. It can be more than one as stated with the typetag. For example /set/instruments ,iss 1 piano guitar drum : in this example integer defines the channel (or instrument set) and following strings defines the instruments*

* this example may be different in real application

MC Conventions

[id]:

id refers to

Player-Based Tracking:

- 0: one player playing; player's data
- 1: two player playing; player-1's data
- 2: two player playing; player-2's data

id defines the playing scheme as well; no other message is sent for 1-player or 2-players playing. Composer should use the id; 0 single player; 1 or 2 multiple player.

Zone-Based tracking

The id of the zone where data was initiated

Movement information sequence:

Activity data is sent when a movement happens AND on change.

Position data is sent with every activity data OR if it has changed.

Location data is sent with every activity data OR if it has changed.

Gestures are sent once when they are recognized.

Gestures do not have any priority above others.

List of Messages:

For the convenience of referencing messages coded and numbered as below

Control SET messages Sxx

Control GET messages Gxx

Tracking messages Txx

CONTROL MESSAGES

There are two types of control messages among modules:

. set: to set a value or a state, etc.

. get: to request data

S01

Request Tracking Data

DESCRIPTION:

ME>MT

ME requests the data it needs to receive or not from TM

ME sends the parent address pattern (see Movement Table) of the data it wants to receive.

examples:

turn on(1) or off(0) all players all tracking data:

/set/alphabet/player/ ,i 1/0

turn on(1) or off(0) player [id]'s all data:

/set/alphabet/player/[id] ,i 1/0

turn on(1) or off(0) player [id]'s all activity data:

/set/alphabet/player/[id]/activity ,i 1/0

turn on(1) or off(0) player [id]'s all activity discrete data:

/set/alphabet/player/[id]/activity/discrete ,i 1/0

turn on(1) or off(0) player [id]'s activity discrete right hand data:

/set/alphabet/player/[id]/activity/discrete/hand/right ,i 1/0

turn on(1) or off(0) all zone(s) data:

/set/alphabet/zone ,i 1/0

turn on(1) or off(0) zone [id]'s all data:

/set/alphabet/zone/[id] ,i 1/0

ADDRESS PATTERN

/set/alphabet/[pattern]

TYPE TAG:

,i

VALUE:

1: True, Turn on (start tracking)

0: False, Turn off (stop tracking)

S02

Start/Stop Tracking Player/Zone

DESCRIPTION:

CM>MT

Select/Deselect player or zone to get TM track player or zone and send data to ME (See TM for the details of auto or user selection process)

ADDRESS PATTERN

/set/ player /[id]/ tracking ,i 1/0

/set/zone/[id]/ tracking ,i 1/0

TYPE TAG:

,i

VALUE:

1: (True) Start tracking

0: (False) Stop tracking

S03

Swap id's

DESCRIPTION:

CM>MT

Swaps players or zones id's. If there are more than two zones push the zones up and first goes to end.

ADDRESS PATTERN

/set/player/swap

/set/zone/swap

TYPE TAG:

NA

VALUE:

none

S04

Initialize:

DESCRIPTION:

CM>ME

Initialize the ME settings to default.

ADDRESS PATTERN:

/set/initialize

TYPE TAG:

N.A.

VALUE:

N.A.

S05

Volume adjustment:

DESCRIPTION:

CM>ME

Set volume to a value.

ADDRESS PATTERN:

/set/volume f

TYPE TAG:

,f

VALUE:

Normalized value to set the volume

S06

Set the List of Soundbanks:

DESCRIPTION:

ME→CM

ME sends the list of soundbanks to GUI in response to Get the List of Soundbanks message.

If ME has only one soundbank, it is optional to use [#]

[#] is the soundbank's number, starts from 0 and can normally be 2 but more is possible for future MEs

RECEIVING MES:

TECHNO

ADDRESS PATTERN:

/set/player/[id]/soundbank/[#]/lists

/set/zone/[id]/soundbank/[#]/lists

TYPE TAG:

,s (as many as the soundbanks)

VALUE:

Name: s, string; soundbank's name

S07

Select Soundbank:

DESCRIPTION:

CM>ME

Selects the soundbank.

RECEIVING MES:

Techno, Particles

ADDRESS PATTERN:

/set/player/[id]/soundbank/[#]/sound

/set/zone/[id]/soundbank/[#]/sound

TYPE TAG:

,i

VALUE:

i, integer: sound index

S08

Set Evolving:

DESCRIPTION:

CM>ME

Set evolving on and off.

RECEIVING MES:

TECHNO

ADDRESS PATTERN:

/set/evolving

TYPE TAG:

,i

VALUE:

1: (True) Turn On

0: (False) Turn Off

S09

Start/Stop All Tracking Messages:

DESCRIPTION:

CM>TM

Set TM to start or stop all tracking messages

RECEIVING MES:

all

ADDRESS PATTERN:

/set/tracking ,l 1/0

TYPE TAG:

,i

VALUE:

1: start sending tracking messages

0: stop sending tracking messages

S10

Loaded:

DESCRIPTION:

ME>CM

ME is loaded.

ADDRESS PATTERN:

/set/loaded

TYPE TAG:

NA

VALUE:

NA

S11

Start/Stop:

DESCRIPTION:

CM>TM

Set sensitivity to given value.

ADDRESS PATTERN:

/set/play

TYPE TAG:

,i

VALUE:

1 (True): Start

0 (False): Stop

S12

Sensitivity adjustment:

DESCRIPTION:

CM>TM

Set sensitivity to given value.

ADDRESS PATTERN:

/set/sensitivity

TYPE TAG:

,f

VALUE:

Normalized value to set the volume

S13

CONFIDENTIAL

Set Blob

DESCRIPTION:

MT>CM(GUI)

Sends shape of blobs of player(s) for GUI.

ADDRESS PATTERN

/set/player/[id]/blob/ ,b

/set/zone/[id]/blob ,b

TYPE TAG:

,b

VALUE:

b: OSC-blob; Blob coordinates

S14

Set Zone Order

DESCRIPTION:

CM>MT

Sets tracking zones order 1 up or down

ADDRESS PATTERN

/set /zone/[id] /order/up

/set/zone/[id] /order/down

TYPE TAG:

NA

VALUE:

none

S15

Set Zone Blob

DESCRIPTION:

CM>TM

ADDRESS PATTERN

/set/zone/[id]/blob ,iiii

TYPE TAG:

,iiii

VALUE:

x1,y1: topleft coordinates of the blob rectangle

x2,y2: bottomright coordinates of the blob rectangle

S16

ME ready:

DESCRIPTION:

ME>CM

Ready message sent on ME's successful initialization

RECEIVING MEs:

all

ADDRESS PATTERN:

/set/ME/ready

TYPE TAG:

NA

VALUE:

NA

S17

TM ready:

DESCRIPTION:

ME>CM

Ready message sent on TM's successful initialization

RECEIVING MEs:

all

ADDRESS PATTERN:

/set/TM/ready

TYPE TAG:

NA

VALUE:

NA

S18

Mute player or zone:

DESCRIPTION:

CM>ME

Mutes the player of zone based on id

RECEIVING MEs:

all

ADDRESS PATTERN:

/set/player/[id]/mute

/set/zone/[id]/mute

TYPE TAG:

,i

VALUE:

1: mutes

0: activates again

S19

Set in-space/in-place playing mode:

DESCRIPTION:

CM>ME

sets in-place playing (for stationary players; vertical playing, etc) or back to in-space playing

RECEIVING MEs:

all

ADDRESS PATTERN:

/set/player/[id]/inPlace

TYPE TAG:

,i

VALUE:

1: sets in-place playing (for stationary players; vertical playing, etc)

0: sets back to in-space playing

S19

Set ERROR:

DESCRIPTION:

any>CM

sets ERROR code in case of ERROR

ADDRESS PATTERN:

/set/error

TYPE TAG:

,i

VALUE:

i: error code

S21

Set blob:

DESCRIPTION:

TM>CM

sets blob coordinates of the blob [id]

ADDRESS PATTERN:

/set/blob/[id]

TYPE TAG:

,iiiiiii

VALUE:

x1,y1,x2,y2,x3,y3...: the vertexes forms the blob

S22

Set player to blob connection:

DESCRIPTION:
CM>TM
Assign a blob to a player
ADDRESS PATTERN:
/set/player/[id]/blob
TYPE TAG:
,i
VALUE:
I: blob id

MOVEMENT TRACKING DATA:

MT → ME

T01

Activity Discrete Hand Left

DESCRIPTION:
Discrete movement of left hand.
ADDRESS PATTERN:
/player/[id]/activity/discrete/hand/left
TYPE TAG:
NA
VALUE:
none

T02

Activity Discrete Hand Right

DESCRIPTION:
Discrete movement of righ hand.
ADDRESS PATTERN:
/player/[id]/activity/discrete/hand/right
TYPE TAG:
NA
VALUE:
none

T03

Activity Discrete Head

DESCRIPTION:
Discrete movement of head.
ADDRESS PATTERN
/player/[id]/activity/discrete/head
TYPE TAG:
NA
VALUE:
none

T04

Activity Discrete Leg Left

DESCRIPTION:
Discrete movement of left leg.
ADDRESS PATTERN
/player/[id]/activity/discrete/leg/left
TYPE TAG:
NA
VALUE:
none

T05

Activity Discrete Leg Right

CONFIDENTIAL

DESCRIPTION:

Discrete movement of right leg.

ADDRESS PATTERN

/player/[id]/activity/discrete/leg/right

TYPE TAG:

NA

VALUE:

None

T06

Activity Discrete Body Upper

DESCRIPTION:

Discrete movement of upper body.

ADDRESS PATTERN

/player/[id]/activity/discrete/body/upper

TYPE TAG:

NA

VALUE:

None

T07

Activity Discrete Body Lower

DESCRIPTION:

Discrete movement of lower body.

ADDRESS PATTERN

/player/[id]/activity/discrete/body/lower

TYPE TAG:

NA

VALUE:

None

T08

Activity Discrete Body Left

DESCRIPTION:

Discrete movement of the left part of body.

ADDRESS PATTERN

/player/[id]/activity/discrete/body/left

TYPE TAG:

NA

VALUE:

None

T09

Activity Discrete Body Right

DESCRIPTION:

Discrete movement of the right part of body.

ADDRESS PATTERN

/player/[id]/activity/discrete/body/right

TYPE TAG:

NA

VALUE:

None

T10

Activity Normal Hand Left

DESCRIPTION:

Normal movement of left hand.

ADDRESS PATTERN

/player/[id]/activity/normal/hand/left

TYPE TAG:

NA

VALUE:

None

T11

Activity Normal Hand Right

DESCRIPTION:

Normal movement of right hand.

ADDRESS PATTERN

/player/[id]/activity/normal/hand/right

TYPE TAG:

NA

VALUE:

None

T12

Activity Normal Head

DESCRIPTION:

Normal movement of head.

ADDRESS PATTERN

/player/[id]/activity/normal/head

TYPE TAG:

NA

VALUE:

None

T13

Activity Normal Leg Left

DESCRIPTION:

Normal movement of left leg.

ADDRESS PATTERN

/player/[id]/activity/normal/leg/left

TYPE TAG:

NA

VALUE:

None

T14

Activity Normal Leg Right

DESCRIPTION:

Normal movement of right leg.

ADDRESS PATTERN

/player/[id]/activity/normal/leg/right

TYPE TAG:

NA

VALUE:

None

T15

Activity Normal Body Upper

DESCRIPTION:

Normal movement of upper body.

ADDRESS PATTERN

/player/[id]/activity/normal/body/upper

TYPE TAG:

NA

VALUE:

None

T16

Activity Normal Body Lower

DESCRIPTION:

Normal movement of lower body.

ADDRESS PATTERN

/player/[id]/activity/normal/body/lower

TYPE TAG:
NA
VALUE:
None

T17

Activity Normal Body Left

DESCRIPTION:
Normal movement of full body.
ADDRESS PATTERN
/player/[id]/activity/normal/body/left
TYPE TAG:
NA
VALUE:
None

T18

Activity Normal Body Right

DESCRIPTION:
Normal movement of full body.
ADDRESS PATTERN
/player/[id]/activity/normal/body/right
TYPE TAG:
NA
VALUE:
None

T19

Peak

DESCRIPTION:
Very large movement with full body.
ADDRESS PATTERN
/player/[id]/activity/peak
TYPE TAG:
NA
VALUE:
None

T20-27

Flow

DESCRIPTION:
Direction of the movement. (UNDER DISCUSSION)
ADDRESS PATTERN
/player/[id]/flow/
TYPE TAG:
NA
VALUE:
None

T28

Location Ready:

DESCRIPTION:
Sent at the beginning when player enters stage and get center-X stable.
ADDRESS PATTERN:
/player/[id]/location/ready
TYPE TAG:
NA
VALUE:
NA

T29

Location Player Present:

DESCRIPTION:

Sent at the beginning when player enters stage.

ADDRESS PATTERN:

/player/[id]/location/present

TYPE TAG:

NA

VALUE:

NA

T30

Location Center-X:

DESCRIPTION:

The midline of the body in X-axis (perpendicular to the camera axis); on plane x.

ADDRESS PATTERN:

/player/[id]/location/centerX

TYPE TAG:

,f

VALUE:

position mapped between 0 and 1 = WIDTH (see: constants)

T31

Location Center-Z:

DESCRIPTION:

The midline of the body in Z axis; movement in the depth of the assumed stage; on plane z

ADDRESS PATTERN:

/player/[id]/location/centerZ

TYPE TAG:

,f

VALUE:

position mapped between 0 and 1 = DEPTH (see: constants)

T32

Location Out Of Range

DESCRIPTION:

This is mainly used to compensate the image for the changing size of the players body. That is, when the player moves close to the camera (e.g. 2 meters away), it should have roughly the same effect as when they are far away (e.g. 5 meters away).

ADDRESS PATTERN:

/player/[id]/location/outOfRange

TYPE TAG:

,string

VALUE:

tooFar: too far from the camera (over 5m)

tooClose: too close (less than 2 m)

T33

Position Height:

DESCRIPTION:

The highest point on the body.

ADDRESS PATTERN:

/player/[id]/position/height

TYPE TAG:

,f

VALUE:

Height; The highest point on the body

T24

Position Height-level:

DESCRIPTION:

Height-level concerns posture. It is divided into 4 levels; thus delivers an integer value of 0,1,2 or 3.

ADDRESS PATTERN:

/player/[id]/position/heightLevel

TYPE TAG:

,i

VALUE:

height level : between 0 to 3 (see description)

T35

Position Vertical Hand Left

DESCRIPTION:

Left hand's vertical position. Many persons with Cerebral Palsy ("Kinderlähmung") cannot extend their arms well, so we want this to also work for arms that are not well-extended, for example, using elbows.

ADDRESS PATTERN:

/player/[id]/position/vertical/hand/left/

TYPE TAG:

,f

VALUE:

Height of the hand

T36

Position Vertical Hand Right

DESCRIPTION:

Right hand's vertical position. Many persons with Cerebral Palsy ("Kinderlähmung") cannot extend their arms well, so we want this to also work for arms that are not well-extended, for example, using elbows.

ADDRESS PATTERN:

/player/[id]/position/vertical/hand/right

TYPE TAG:

,f

VALUE:

Height of the hand

T37

Position Side Hand Left

DESCRIPTION:

The distance of the left hand from the side of the body; in plane x.

ADDRESS PATTERN:

/player/[id]/position/side/hand/left

TYPE TAG:

,f

VALUE:

The distance <how to calculate: when it is 1 when it is 0=>

T38

Position Side Hand Right

DESCRIPTION:

The distance of the right hand from the side of the body; in plane x.

ADDRESS PATTERN:

/player/[id]/position/side/hand/right

TYPE TAG:

,f

VALUE:

The distance <how to calculate: when it is 1 when it is 0=>

T39

Position Side Foot Left

DESCRIPTION:

The distance of the left foot from the side of the body; in plane x.

ADDRESS PATTERN:

/player/[id]/position/side/foot/left

TYPE TAG:

,f

VALUE:

The distance <how to calculate: when it is 1 when it is 0=>

T40

Position Side Foot Right

DESCRIPTION:

The distance of the right foot from the side of the body; in plane x.

ADDRESS PATTERN:

/player/[id]/position/side/foot/right

TYPE TAG:

,f

VALUE:

The distance <how to calculate: when it is 1 when it is 0=>

T41

Position Front Hand Left

DESCRIPTION:

The distance of the hand from the front of the body; in plane z.

ADDRESS PATTERN:

/player/[id]/position/front/hand/left

TYPE TAG:

,f

VALUE:

The distance of the left hand from the body in the front direction
<how to calculate: when it is 1 when it is 0=>

T42

Position Front Hand Right

DESCRIPTION:

The distance of the right hand from the front of the body; in plane z.

ADDRESS PATTERN:

/player/[id]/position/front/hand/right

TYPE TAG:

,f

VALUE:

The distance of the hand from the body in the front direction
<how to calculate: when it is 1 when it is 0=>

T43

Position Front Foot Left

DESCRIPTION:

The distance of the foot from the front of the body; in plane z.

ADDRESS PATTERN:

/player/[id]/position/front/foot/left

TYPE TAG:

,f

VALUE:

The distance of the left foot from the body in the front direction
<how to calculate: when it is 1 when it is 0=>

T44

Position Front Foot Right

DESCRIPTION:

The distance of the right foot from the front of the body; in plane z.

ADDRESS PATTERN:

/player/[id]/position/front/foot/right

TYPE TAG:

,f

VALUE:

The distance of the foot from the body in the front direction
<how to calculate: when it is 1 when it is 0=>

T45

Position Width:

DESCRIPTION:

Width is the distance between the left-most and right-most points on the body, i.e. the player's total width, changing continuously. Actually refers to the blobs maximum width.

ADDRESS PATTERN:

/player/[id]/position/width

TYPE TAG:

,f

VALUE:

width (f in reference to the Width or to its own maximum that changes)

T46

Gesture Hit Overhead

DESCRIPTION:

Hands outward-from-the-body movements of hands to overhead discretely.

ADDRESS PATTERN:

/player/[id]/gesture/hit/overHead

TYPE TAG:

NA

VALUE:

None

T47

Gesture Hit Side Left

DESCRIPTION:

Left hand goes left discretely.

ADDRESS PATTERN:

/player/[id]/gesture/hit/side/left

TYPE TAG:

NA

VALUE:

None

T48

Gesture Hit Side Right

DESCRIPTION:

Right hand goes right discretely.

ADDRESS PATTERN:

/player/[id]/gesture/hit/side/right

TYPE TAG:

NA

VALUE:

None

T49

Gesture Hit Downward Left

DESCRIPTION:

Left hand goes downward discretely.

ADDRESS PATTERN:

/player/[id]/gesture/hit/down/left

TYPE TAG:

NA

VALUE:

None

T50

Gesture Hit Downward Right

DESCRIPTION:

Right hand goes downward discretely.

ADDRESS PATTERN:

/player/[id]/gesture/hit/down/right
TYPE TAG:
NA
VALUE:
None

T51

Gesture Hit Forward Left

DESCRIPTION:
Left hand goes forward towards the camera discretely.
ADDRESS PATTERN:
/player/[id]/gesture/hit/forward/left
TYPE TAG:
NA
VALUE:
None

T52

Gesture Hit Forward Right

DESCRIPTION:
Right hand goes forward towards the camera discretely.
ADDRESS PATTERN:
/player/[id]/gesture/hit/forward/right
TYPE TAG:
NA
VALUE:
None

T53

Gesture Kick Side Left

DESCRIPTION:
Left foot goes sides discretely.
ADDRESS PATTERN:
/player/[id]/gesture/kick/side/left
TYPE TAG:
NA
VALUE:
None

T54

Gesture Kick Side Right

DESCRIPTION:
Right foot goes sides discretely.
ADDRESS PATTERN:
/player/[id]/gesture/kick/side/right
TYPE TAG:
NA
VALUE:
None

T55

Gesture Kick Forward Left

DESCRIPTION:
Left foot goes forward towards the camera discretely.
ADDRESS PATTERN:
/player/[id]/gesture/kick/forward/left
TYPE TAG:
NA
VALUE:
None

T56

Gesture Kick Forward Right

DESCRIPTION:

Right foot goes forward towards the camera discretely.

ADDRESS PATTERN:

/player/[id]/gesture/kick/forward/right

TYPE TAG:

NA

VALUE:

None

T57

Gesture Double-Arm Side (DAS):

DESCRIPTION:

Opening both arms quickly to the side (opening the arms slowly is not DAS).

ADDRESS PATTERN:

/player/[id]/gesture/doubleArmSide

TYPE TAG:

NA

VALUE:

None

T58

Gesture Double-Arm Side Close:

DESCRIPTION:

Closing both arms quickly to the centre (closing the arms slowly is not DAS).

ADDRESS PATTERN:

/player/[id]/gesture/doubleArmSideClose

TYPE TAG:

NA

VALUE:

None

T59

Gesture Jump:

DESCRIPTION:

Both feet have to come off the floor by ca. 10 cm.

ADDRESS PATTERN:

/player/[id]/gesture/jump

TYPE TAG:

NA

VALUE:

NA

T60

Gesture Clap:

DESCRIPTION:

Bringing hands together.

ADDRESS PATTERN:

/player/[id]/gesture/clap

TYPE TAG:

NA

VALUE:

NA

T61

Zone Discrete:

DESCRIPTION:

Discrete movement in zone

ADDRESS PATTERN:

/zone/[id]/activity/discrete

TYPE TAG:

NA

VALUE:
NA

T62

Zone Normal:

DESCRIPTION:

Discrete movement in zone

ADDRESS PATTERN:

/zone/[id]/activity/normal

TYPE TAG:

,f

VALUE:
Activity level

T63-66

Zone Flow:

DESCRIPTION:

Discrete movement in zone (UNDER DISCUSSION)

ADDRESS PATTERN:

/zone/[id]/activity/flow

TYPE TAG:

,f

VALUE:
Activity direction

MUSIC ENVIRONMENTS

Introduction

The Musical environments (MEs) are the creative modules of MC. Each ME is a stand-alone program, which is loaded and started by the MC. Once running, the ME communicates with the MC through APIs. The APIs are in principle common to all MEs, though some may have additional parameters specific to the ME.

The MEs receive OSC data streams, generated by the TM, and process them to create the designated audio output. The data streams are based on a Movement Alphabet (MA) which provides a standardized movement-to-data formula.

There are six ME's:

1. Techno
2. Tonality
3. Fields
4. Particles
5. Drums
6. Import-your-own-music (temporary name)

Conceptual Requirements (including notes to composers)

All ME's Should:

1. Promote Inclusion
2. Be easy to play. Super easy.
3. Embodiment a musical concept and body-sound relationship
4. 1- and 2-person playing (though not every ME has 2-person tracking)
5. Offer a variety of sounds
6. Give the player a tangible role in the experience
7. Compatible with our API
8. Offer different modes of Use, e.g. one, two player(s) or zones-only
9. Have an evolving mode option (so it changes by itself, over time)

Note to Composers 1: Causality

Making music from movement is easy. The hard part is to achieve this in a tangible way – so that the player understands their role in the music-making process. Here are some tricks to making clear causality:

1. Scratchy wind – When noise, or white noise (like the sound of wind) is controlled by activity (higher activity = higher loudness and higher pitch) it results in one of the strongest most universal mappings we have found. (in the Techno ME, we add a little noise to the activity scrubbing).
2. Hits, kicks, punches, etc. – hands side, overhead, punches, kicks. Double arm side. Jump... When used for accents is highly convincing.
3. Linear sequences – e.g. Tonality, In-Space, when you walk across the room, the notes get higher and higher. This is very tangible and helpful for beginners.
4. Pitch bend with height – Almost every ME 'melts' when you go towards the floor.
5. Discretes – this would be number 1, except for one thing: people do not hold still unless you ask them to. And even then... some do not. Still, once you learn to do it, its highly convincing and gentle at the same time.

Note to Composers 2: 1- and 2-player schemes

For the composer, implementing 2-person playing is a challenge: The 2-parts need to be quite different, otherwise the players will get mixed up who is making which sound. Also, this probably means that the density of sounds from each player should be reduced — but hey, I’m not a composer! Note: Single-player mode, and one-half of a 2-player scheme are not the same thing. (Ask me if this is not clear).

Note: we were going to make all MEs 2-player able. Maybe one day, but now we decided that for the 3.0:

only Drums, Tonality, Fields will offer 2-player mode. Particles is still a maybe – though the tests at Fraunhofer sort of said that it was not worth the effort.

Note to Composers 3: Initialization and GUI elements

During the initialization phase MEs should send “/set/player/[id]/soundbank/[#]/list” message to set the instruments. This gives flexibility to the authors for later choices. The other options are already fixed by design. If you have other similar suggestions, please inform us.

Please inform us about any GUI elements that seem missing, or mismatched with your current or past work.

Note to Composers 4: Evolving mode

The idea of evolving mode came from a therapist for severely disabled children. He told us his kids are left alone for hours at a time, so that a system that changes by itself over time would be nice.

All kinds of changes might can occur, but what comes to mind first is that the instrumentation (or sound bank) changes automatically, over time. The rate of evolution or time span might be “loop after 10mins”, up to “loop after 1hour”, for example. Or random. Or at least which sound comes first could

Note to Composers 5: hL0 and hL3

As described above (Fig. 11), heightLevel (hL) is a Movement Alphabet feature that refers to the posture of the player. If they are standing, then they are in heightLevel2, reaching overhead is hL3. Crouching low is hL1, and being on the floor is hL0.

Most MEs use hL0 to bring the player into a crazy, abstract “melted” version of the ME. Quite far from the original, but still referring to it in someway. Activity and height can be used to modulate it.

hL3, is either higher frequencies, or high-pass filter, or it brings in a special high-pitched track. (Tonality-In-space uses it to play a second instrument).

Tracking Modes

From the perspective of the tracking, and therefore the data being sent to the ME’s, there are three modes:

1-player, 2-player, and Zones-only

If the ME receives a movement message with the player id ‘0’ (e.g. **/player/0/activity/normal 0,72**), it means that only one player is being tracked. Thus, it is in “1-player mode” and should play accordingly.

If the ME receives a movement message with player id ‘1’ or ‘2’ (e.g. **/player/1/activity/normal 0,72**), it means there are two players being tracked, and thus it is “2-player mode” and should play accordingly.

If the ME receives a movement message with ‘zone’ (e.g. **/zones/0/activity/normal 0,72**), it means it is in “zone mode” (formerly called “bed”) and should play accordingly.

Composers should think in a paradigm in which any message can come at anytime and perform its task regardless of tracking mode.

User Modes

Although the tracking modes (described above) are a universal concept, in practice not every ME offers this selection. I.e. the user experience, is somewhat different. First of all, 2-player mode is not available to every ME (at least not in the current MC3.0). Moreover, due to the fact that there are fundamental differences in the mappings of each ME, different options are presented to the user for each ME.

In-Place, In-Space

Tonality and Particles offer a choice of “In-space” and “In-place” (we may change these names...):

- **In-space** – choosing sounds, or notes, by moving through the room, /player/0/location/centerX 0,72.
- **In-place** – choosing sounds, or notes, by height -- either normal overall height, or hand height. If hand height is used, then there are two choosers (left and right hand height, in other words, /player/0/position/vertical/hand/left 0,72 and /player/0/position/vertical/hand/right 0,23, etc.)

Accents

Many ME’s offer accents.

Tonality – chords usually. possibly other instruments

Techno – yes

Particles – yes, though it has not been implemented

Fields – yes, as an option

Importer – not sure yet.

Drums – only overhead.

Each accents is not a single sound, but a collection of similar samples, ca. 7-10. They are “naturalized”, so that even if you hear the same exact sample twice, it will always sound a little different.

They need to contrast strongly with the other notes being played – they are not played very often.

Some sounds, such as vocalizations (shouts especially), animal sounds, or some kinds of percussion, are atonal – and will work well with any key.

For Importer there is the challenge of how to find accents that fit universally. Maybe the key can be analyzed, and accents generated which fit the key. Or, this may not be necessary.

Choosing to use different gestures to trigger the accents is important since abilities vary.

Musical controls
...
ACCENTS:
Jump <input type="checkbox"/>
Kick <input type="checkbox"/>
Overhead <input type="checkbox"/>
Peak <input type="checkbox"/>
Meltdown <input type="checkbox"/>
All Accents active / All accents in-active <input type="checkbox"/>

GUI Elements

Each ME has its own GUI. The “General Controls” are common to all ME’s. Additionally, there are options that are similar, but not identical, and others that are unique to a certain ME.

Typical GUI controls (see individual MEs for more examples):

General controls	Player controls	Musical controls
Start (button)	1 Player / 2 Players / Zones (radio button)	Scale (dropdown list)
Stop (button)		Direction reverse (checkbox)
Volume (slider)	For each player and zone: Play in place / play in space (radio button)	
Sensitivity (slider)	Instrument 1 (dropdown list) Instrument 2 (dropdown list)	

ME Technical Requirements

Development Platforms

Currently MC handles communication with

Pure Data

SuperCollider

CSound

But other environments can also be added

Tracking Schemes

TM Tracking Scheme uses two separate approaches:

“tracking player”, TM tracks one or two player(s) and recognizes the body and its parts;

“tracking zones” TM does not try to recognize the body or its parts, but only tracks the movement in the selected areas.

Messages

The ME receives messages from two sources:

- TM, via the Player’s movements
- CM via operator’s control actions

Techno

Note to Composers

- We made a separate document for the music tasks of each ME. Look in the folder “MEWork” and you will see your tasks, and other information, up-to-date.

Description

More than any other ME, Techno is concerned with pop music. I.e. the user-experience begins with a more-or-less finished song, and the user is given ways to join in and add to it. True to its name, the beat never stops.

Modes of Use

Features

Techno offer these ways to control the music:

1. Levels, based on movement activity level

- 1 no user present
- 2 user is present
- 3 continuous transformation from small dancing to large dancing
- 4 we are experimenting with activity scratching (adding noise etc. to activity normal). wind, water sounds...

2. Hits, to the side, or front

Plays individual notes or chords in a melody. (We experimented with Left being different from Right, but decided it is too complicated.)

3. Breaks

We came up with 2 ways basic ways to do it. (Either way discourages movement, so the usual pumped-up sound needs to be replaced with something of equal power).

- 1 extension of arms to the side (cross-fade to break music, or as in ML's case, a kind swimming comes in)
- 2 height level changes
 - 2a Overhead – something high, and/or high-pass filter
 - 2b Crouch – base break
 - 2c Lying on floor – something totally crazy (like granular synthesis. Does not need a beat)

4. Accents, probably not quantized

- kicks
- jumps
- peak
- button in tablet controller - so Operator can play along.

5. 2-Person play

In-place

for MC4.0 (this would be like nagual... we can talk about it, but I think you'll agree, its too soon for the

3.0)

Alphabet Used (marked in red)

				Techno Modes		
				1-Person In-place	1-zone	2-zones
					this would be something new... I am not sure it has any value (let's talk about it).	this is the normal techno 'bed'.
/activity	/discrete	/hand	/left			
			/right			
		/head	/left			
			/right			
		/leg	/left			
			/right			
		/body	/upper			
			/lower			
			/right			
			/left			
	/normal	/hand	/left	energy level and noise level (in some cases)		
			/right			
		/head	/left			
			/right			
		/leg	/left			
			/right			
		/body	/upper			
			/lower			
		/body	/right			
			/left			

	/peak			not currently used, except in ML's beat doubling effect. or other effect (wild African drums).		
	/flow	/leftwards	/left			
			/right			
		/rightwards	/left			
			/right			
		/upwards	/left			
			/right			
		/downwards	/left			
			/right			
/location	/ready			turns on the normal activity response (see above) so that the player can pump up the music, play hits, etc.		
	/present			music goes from "no user present" to "user present", (but do not yet activate /activity etc. until /ready.)		
	/centerX					
	/centerZ					
	/outOfRange					
/position	/height			Used within hL3. at maximum, we talked about 'super angels'. could be used with hL0 and 1?		
	/heightLevel			0- granulated stuff (it can be quite crazy (ML) 1- certain instruments only 3- certain instruments (e.g. angels)		
	/vertical	/hand	/left			
			/right			
	/side	/hand	/left	sometimes used		
			/right	sometimes used		
		/foot	/left			
			/right			
	/front	/hand	/left	sometimes used		
			/right	sometimes used		
		/foot	/left			
			/right			
	/width			sometimes used		
/gesture	/hit	/overhead		maybe an accent like a bell, but not if it interferes with hL3.		
		/side	/left	melody note (iteration)		
			/right	melody note (iteration)		
		/down	/left	alternative to /side		
			/right	alternative to /side		
		/forward	/left	alternative to /side		

			/right	alternative to /side		
	/kick	/side	/left	accent*		
			/right	accent*		
		/forward	/left	accent*		
			/right	accent*		
	/doubleArmSide			accent*		
	/doubleArmSideClose					
	/jump			accent*		
	/clap					
/activity	/normal					zone1: higher frequencies zone2: lower frequencies, or other strong difference (dividing the instruments is usually better than filtering!)
		/discrete				
		/flow	/leftwards			
			/rightwards			
			/upwards			
			/downwards			

* - each could be a different sound, but that sounds like nuts. more probably some simpler scheme, such as upper body this, lower body that, or purely random, sequenced or pseudo-random. This – along with many other details – is discussed in the Techno Music tasks document in the MEWork folder.

List of Messages (MC 2.0 > 3.0)

MC 2.0			MC 3.0		
Message Pattern	Typetag	Mode (R,C,B)	Message Pattern		Typetag
TECHNO SENDs on ports 9085(R) / 9986(C)			TECHNO SENDs to CM on port 6065		
			/set/loaded		None
/ready	1	R,C,B	/set/ready		None
			/set/player/[id]/soundbank/[#]/list	,s	String(s)
			TECHNO SENDs to TM on port 6061		
			/set/alphabet/[pattern]	,i	1/0
			examples*: /set/alphabet/player/[id]/activity/discrete /set/alphabet/player/[id] /set/alphabet/player/[id]/activity *: see OSC messages section for more		
TECHNO RECEIVES on ports 7044(R) / 7048(C) / 7888(B)			TECHNO RECEIVES from CM on port 6560		

/start	i	R,C,B	/set/play	,i	1
/stop	i	R,C,B			
/volume	,f	R,C,B	/set/volume	,f	volume
			/set/sensitivity	,f	sensitivity
			/set/player/[id]/tracking /set/zone/[id]/tracking	,i	1/0
/soundbank	i	R,C,B	/set/player/[id]/soundbank/[#]	,i	Instrument's index
TECHNO RECEIVES from TM on port 6160					
			/player/[id]/activity/discrete/hand/left		None
			/player/[id]/activity/discrete/hand/right		None
			/player/[id]/activity/discrete/head		None
			/player/[id]/activity/discrete/leg/left		None
			/player/[id]/activity/discrete/leg/right		None
			/player/[id]/activity/discrete/body/upper		None
			/player/[id]/activity/discrete/body/lower		None
			/player/[id]/activity/discrete/body/right		None
			/player/[id]/activity/discrete/body/left		None
/armSideLeft	f	C	/player/[id]/activity/normal/hand/left	,f	Norm.
/armSideRight	f	C	/player/[id]/activity/normal/hand/right	,f	Norm.
			/player/[id]/activity/normal/head	,f	Norm.
			/player/[id]/activity/normal/leg/left	,f	Norm.
			/player/[id]/activity/normal/leg/right	,f	Norm.
/activityNormal1	f	R	/player/[id]/activity/normal/body/upper	,f	Norm.
/activityNormal2	f	C,B	/player/[id]/activity/normal/body/lower	,f	Norm.
			/player/[id]/activity/normal/body/right	,f	Norm.
			/player/[id]/activity/normal/body/left	,f	Norm.
			/player/[id]/activity/peak	,f	Norm.
			/player/[id]/flow/leftwards/left		None
			/player/[id]/flow/leftwards/right		None
			/player/[id]/flow/rightwards/left		None
			/player/[id]/flow/rightwards/right		None
			/player/[id]/flow/upwards/left		None
			/player/[id]/flow/upwards/right		None
			/player/[id]/flow/downwards/left		None
			/player/[id]/flow/downwards/right		None
			/player/[id]/location/ready	,i	0/1
			/player/[id]/location/present	,i	0/1
			/player/[id]/location/centerX	,f	Norm.
			/player/[id]/location/centerZ	,f	Norm.
			/player/[id]/location/outOfRange		None
/height1	f	R	/player/[id]/position/height	,f	Norm.
/heightLevel1	i	R	/player/[id]/position/heightLevel	,i	0,1,2,3
			/player/[id]/position/vertical/hand/left	,f	Norm.
			/player/[id]/position/vertical/hand/right	,f	Norm.

			/player/[id]/position/side/hand/left	,f	Norm.
			/player/[id]/position/side/hand/right	,f	Norm.
			/player/[id]/position/side/foot/left	,f	Norm.
			/player/[id]/position/side/foot/right	,f	Norm.
			/player/[id]/position/front/hand/left	,f	Norm.
			/player/[id]/position/front/hand/right	,f	Norm.
			/player/[id]/position/front/foot/left	,f	Norm.
			/player/[id]/position/front/foot/right	,f	Norm.
/width1	f	R,C	/player/[id]/position/width	,f	Norm.
			/player/[id]/gesture/hit/overhead		None
			/player/[id]/gesture/hit/side/left		None
			/player/[id]/gesture/hit/side/right		None
			/player/[id]/gesture/hit/down/left		None
			/player/[id]/gesture/hit/down/right		None
			/player/[id]/gesture/hit/forward/left		None
			/player/[id]/gesture/hit/ forward /right		None
			/player/[id]/gesture/kick/side/left		None
			/player/[id]/gesture/kick/side/right		None
			/player/[id]/gesture/kick/forward/left		None
			/player/[id]/gesture/kick/forward/right		None
			/player/[id]/gesture/doubleArmSide		None
			/player/[id]/gesture/doubleArmSideClose		None
			/player/[id]/gesture/jump		None
			/zone/[id]/activity/discrete		None
			/zone/[id]/activity/normal	,f	Norm.
			/zone/[id]/activity/flow/leftwards		None
			/zone/[id]/activity/flow/rightwards		None
			/zone/[id]/activity/flow/upwards		None
			/zone/[id]/activity/flow/downwards		None
			/zone/[id]/activity/flow/forward		None

GUI Elements

General controls	Player controls	Musical controls
Start (button)	(radio button in -place)	Evolves Over Time (checkbox)
Stop (button)	Zones	Break (push button)
Volume (slider)	Sound bank (radio button)	n x Accent (push b) *
Sensitivity (slider)	<ul style="list-style-type: none"> Groove 1 "She" Groove 2 "Cat" Groove 3 "find a name you like" Groove 4 "find a name you like" 	

*: these are the buttons that let the operator "play along with the player" by tapping on the tablet.

Future market?

Techno, the music genre, is, of course, huge. Maybe we can sell the MC to other markets? Imagine, e.g., after we build some good techno MEs, that we talk to pop stars about contributing songs à la Guitar Hero.

Fields

Note to Composers

- We made a separate document for the music tasks of each ME. Look in the folder “MEWork” and you will see your tasks, and other information, up-to-date.

Description

Simple: you move, you hear. Highly intuitive, if not always musical. We could make different body parts create different sounds, but we don’t want to... This introduces a level of complexity that we do not want (at least not in the primary product). Such goes against the basic metaphor: you are the bird, etc.

Features

Zones Mode

- Discretes -- Small isolated sounds. example: a single chirp for bird
- Activity Normal-- example: low activity = a singing bird, high activity = many birds, and, maybe,
- Activity Flow

In-Place (1-Player or 2-Player)

- in addition to the Zones features just mentioned, In-Place offers:
- Special Accents -- one or two groups of sounds for special gestures -- for example: in rain, you might jump and get thunder
 - Peak
 - Kick (maybe Hit for wheel chair?)
 - Jump*
(*- may not be possible with passive stereovision)
 - Hits, including overhead and double-arm-side could be used
 - Stretching arms wide might be used in Peter’s water to get a whale sound
 - Meltdown when you go to the floor. This is a simple pitch-bend effect.

Narratives

- These are stories, They represent a more holistic approach, based on, for example,
- two environments – like water and air
 - each has some background, and some modulated sounds
 - as above, player can do 1 or 2 special things for a special sound
 - there is a transition between them (we talked about accumulated activity, and also arms overhead for 3 seconds, etc. The two could also be assigned to left zone and right zone.

Alphabet Used (marked in red)

				Fields Modes
--	--	--	--	--------------

				1-Pers. (or2-pers.) Zone / In-place hybrid	1-zone	2-zones
					I'm not sure we need this option (let's talk about it).	this is the normal mode
/activity	/discrete	/hand	/left	discrete sounds		
			/right			
		/head				
		/leg	/left			
			/right			
		/body	/upper			
			/lower			
			/right			
			/left			
	/normal	/hand	/left	energy level		
			/right			
		/head				
		/leg	/left			
			/right			
		/body	/upper			
			/lower			
			/right			
			/left			
	/peak			biggest level sound		
	/flow	/leftwards	/left			
			/right			
		/rightwards	/left			
			/right			
		/upwards	/left			
			/right			
		/downwards	/left			
			/right			
/location	/ready					
	/present					
	/centerX					
	/centerZ					
	/outOfRange					
/position	/height			in hL1 and 0 – could melt		
	/heightLevel			hL0 – could be something special in hL1 and 0 – could be melt using height		
	/vertical	/hand	/left			
			/right			
	/side	/hand	/left			
			/right			
		/foot	/left			
			/right			
	/front	/hand	/left			
			/right			
		/foot	/left			
			/right			
	/width					
/gesture	/hit	/overhead		accent*		
		/side	/left	accent*		
			/right	accent*		

		/down	/left	accent*		
			/right	accent*		
		/forward	/left	accent*		
			/right	accent*		
	/kick	/side	/left	accent*		
			/right	accent*		
		/forward	/left	accent*		
			/right	accent*		
	/doubleArm Side			accent (something very special)		
	/doubleArm SideClose					
/activity	/jump			accent (something very large)		
	/clap					
	/normal				energy level	energy level
	/discrete				discrete sounds	discrete sounds
	/flow	/leftwards				
		/rightwards				
		/upwards				
		/downwards				

* - each could be a different sound, but that sounds like nuts. more probably some simpler scheme, such as upper body this, lower body that, or purely random, sequenced or pseudo-random. This – like many details – is discussed in the Techno Music tasks document in the MEWork folder.

List of Messages Compared in MC 2.0 vs. 3.0

MC 2.0			MC 3.0		
Message Pattern	Typetag	Mode (R,C,B)	Message Pattern	Typetag	Argument
FIELD SENDs on ports 9991(R,B,C)			FIELD SENDs to CM on port 6065		
			/set/loaded		None
			/set/player/[id]/soundbank/[#]/list /set/zone/[id]/soundbank/[#]/list	,s	String(s)
/ready	1	R,C	/set/ready		None
			FIELD SENDs to TM on port 6061		
			/set/alphabet/[pattern]	,i	1/0
			examples*: /set/alphabet/player/activity/discrete /set/alphabet/player/[id] /set/alphabet/player/[id]/activity /set/alphabet/zone/[id]/activity *: see OSC messages section for the rules		
FIELD RECEIVES on ports 7020(R,B,C)			FIELD RECEIVES from CM on port 6560		
/start	i	R	/set/play	,i	1

/stop	i	R			
/volume	,f	R	/set/volume	,f	volume
/sensitivity	f	R	/set/sensitivity	,f	sensitivity
/sound1	i	R	/set/player/[id]/soundbank/[#]	,i	Instrument's index
/sound2	i	R			
			FIELD RECEIVES from TM on port 6160		
/activitySensitive1 /activitySensitive2	f	R	/player/[id]/activity/discrete/hand/left		None
			/player/[id]/activity/discrete/hand/right		None
			/player/[id]/activity/discrete/head		None
			/player/[id]/activity/discrete/leg/left		None
			/player/[id]/activity/discrete/leg/right		None
			/player/[id]/activity/discrete/body/upper		None
			/player/[id]/activity/discrete/body/lower		None
			/player/[id]/activity/discrete/body/right		None
			/player/[id]/activity/discrete/body/left		None
/activityNormal1 /activityNormal2	f	R	/player/[id]/activity/normal/hand/left	,f	Norm.
			/player/[id]/activity/normal/hand/right	,f	Norm.
			/player/[id]/activity/normal/head	,f	Norm.
			/player/[id]/activity/normal/leg/left	,f	Norm.
			/player/[id]/activity/normal/leg/right	,f	Norm.
			/player/[id]/activity/normal/body/upper	,f	Norm.
			/player/[id]/activity/normal/body/lower	,f	Norm.
			/player/[id]/activity/normal/body/right	,f	Norm.
			/player/[id]/activity/normal/body/left	,f	Norm.
			/player/[id]/activity/peak	,f	Norm.
			/player/[id]/flow/leftwards/left		None
			/player/[id]/flow/leftwards/right		None
			/player/[id]/flow/rightwards/left		None
			/player/[id]/flow/rightwards/right		None
			/player/[id]/flow/upwards/left		None
			/player/[id]/flow/upwards/right		None
			/player/[id]/flow/downwards/left		None
			/player/[id]/flow/downwards/right		None
			/player/[id]/location/ready	,i	0/1
			/player/[id]/location/present	,i	0/1
			/player/[id]/location/centerX	,f	Norm.
			/player/[id]/location/centerZ	,f	Norm.
			/player/[id]/location/outOfRange		None
/height1 /height2	f	R	/player/[id]/position/height	,f	Norm.
			/player/[id]/position/heightLevel	,i	0,1,2,3
			/player/[id]/position/vertical/hand/left	,f	Norm.
			/player/[id]/position/vertical/hand/right	,f	Norm.
			/player/[id]/position/side/hand/left	,f	Norm.
			/player/[id]/position/side/hand/right	,f	Norm.
			/player/[id]/position/side/foot/left	,f	Norm.

			/player/[id]/position/side/foot/right	,f	Norm.
			/player/[id]/position/front/hand/left	,f	Norm.
			/player/[id]/position/front/hand/right	,f	Norm.
			/player/[id]/position/front/foot/left	,f	Norm.
			/player/[id]/position/front/foot/right	,f	Norm.
			/player/[id]/position/width	,f	Norm.
			/player/[id]/gesture/hit/overhead		None
			/player/[id]/gesture/hit/side/left		None
			/player/[id]/gesture/hit/side/right		None
			/player/[id]/gesture/hit/down/left		None
			/player/[id]/gesture/hit/down/right		None
			/player/[id]/gesture/hit/forward/left		None
			/player/[id]/gesture/hit/ forward /right		None
			/player/[id]/gesture/kick/side/left		None
			/player/[id]/gesture/kick/side/right		None
			/player/[id]/gesture/kick/forward/left		None
			/player/[id]/gesture/kick/forward/right		None
			/player/[id]/gesture/doubleArmSide		None
			/player/[id]/gesture/doubleArmSideClose		None
			/player/[id]/gesture/jump		None
			/zone/[id]/activity/discrete		None
			/zone/[id]/activity/normal	,f	Norm.
			/zone/[id]/activity/flow/leftwards		None
			/zone/[id]/activity/flow/rightwards		None
			/zone/[id]/activity/flow/upwards		None
			/zone/[id]/activity/flow/downwards		None
			/zone/[id]/activity/flow/forward		None

GUI Elements

General controls	Player controls	Musical controls
Start (button)	1-Player / 2-Player / Zones (radio button)	Evolves Over Time (check box)
Stop (button)		Either:
Volume (slider)	For each Zones	All Accents active (checkbox)
Sensitivity (slider)	soundbank (dropdown list)	or, maybe also individually:
		Jump (checkbox)
		Kick (checkbox)
		Overhead (checkbox)
		Peak (checkbox)
		Meltdown (checkbox)
		Evolving (checkbox)

Tonality

Note to Composers

- We made a separate document for the music tasks of each ME. Look in the folder “MEWork” and you will see your tasks, and other information, up-to-date.

Description

Features

In-Space

Imagine the piano is somehow on the floor of the room, like a cat walking on a piano.

There are seven features:

- notes left to right
- discretes play single notes (sometimes small chords)
- additional chords overhead, kick, double-arms-side, etc.
- das plays chords
- holding arms open, arpeggiates
- going to floor melts down (I think we are going to kill this)
- choice of scales (including the new degree-based scales) Tonicity Modes

Alphabet used (marked in red)

				Tonality Modes			
				1-Pers. (or2-pers.) In-Space	1-Pers. (or2-pers.) In-Place	1-zone	2-zones
						I'm not sure we need this option, but I think so. (let's talk about it).	
/activity	/discrete	/hand	/left	plays single notes and small chords	plays single notes and small chords		
			/right		plays single notes and small chords		
		/head			plays single notes and small chords		
		/leg	/left				
			/right				
		/body	/upper				
			/lower				
			/right		plays single notes and small chords		
			/left		plays single notes and small chords		
	/normal	/hand	/left	plays music	plays music		
			/right		plays music		
		/head			plays music (could be random L/R, or something special)		
		/leg	/left				
			/right				
		/body	/upper				
			/lower				
			/right				
			/left				
		/peak					
	/flow	/leftwards	/left				

			/right				
		/rightwards	/left				
			/right				
		/upwards	/left				
			/right				
		/downwards	/left				
			/right				
/location	/ready				delay playing until player is ready (we need to test this)		
	/present						
	/centerX						
	/centerZ						
	/outOfRange						
/position	/height			in hL1 and 0 – could melt	chooses notes (pitch) when no arms are found*		
	/heightLevel			hL0 – could be something special in hL1 and 0 – could be melt using height hL3 plays overhead instrument. (perhaps note-by-note, like room)			
	/vertical	/hand	/left		chooses notes when arms are found*		
			/right				
	/side	/hand	/left				
			/right				
		/foot	/left				
			/right				
	/front	/hand	/left				
			/right				
		/foot	/left				
			/right				
	/width						
/gesture	/hit	/overhead		plays “overhead instrument” small chords			
		/side	/left	plays chords **			
			/right	plays chords			
		/down	/left	plays chords			
			/right	plays chords			
		/forward	/left	plays chords			
			/right	plays chords			
	/kick	/side	/left	plays chords			
			/right	plays chords			
		/forward	/left	plays chords			
			/right	plays chords			
	/doubleArmSide			plays chords			
	/doubleArmSideClose			following /das, if not received, arpeggiates			
	/jump			plays BIG chord			

	/clap						
/activity	/normal					plays music	strategy for 2zones TBA
	/discrete					plays discretetes	plays discretetes
	/flow	/leftwards					
		/rightwards					
		/upwards					
		/downwards					

* - this needs to be discussed.

** - or other accent (another instrument?). as wi all MEs, gestures do not need to all do something different – but they could. This is simply a strategy decision.

List of Messages Compared in MC 2.0 vs 3.0

MC 2.0			MC 3.0		
Message Pattern	Typetag	Mode (R,C,B)	Message Pattern	Typetag	Argument
TONALITY SENDs on ports 9082(R) / 9986(C)			TONALITY SENDs to CM on port 6065		
			/set/loaded		None
/ready	1	R,C,B	/set/ready		None
			/set/player/[id]/soundbank/[#]/list /set/zone/[id]/soundbank/[#]/list	,s	String(s)
			TONALITY SENDs to TM on port 6061		
			/set/alphabet/[pattern] examples*: /set/alphabet/player/activity/discrete /set/alphabet/player/[id] /set/alphabet/player/[id]/activity /set/alphabet/zone/[id]/activity *: see OSC messages section for the rules	,i	1/0
TONALITY RECEIVES on ports 7056(R) / 7060(C) / 7090(B)			TONALITY RECEIVES from CM on port 6560		
/start	i	R,C,B	/set/play	,i	1
/stop	i	R,C,B			
/volume	,f	R,C,B	/set/volume	,f	volume
/sensitivity	,f	R,C,B	/set/sensitivity	,f	sensitivity
/key	i	R	/set/key	,s	Key name
/instrument_1	i	R	/set/player/[id]/soundbank/[#]	,i	Instrument's index
/instrument_2	i	R			
/instrument_r	i	C,B			
/instrument_l	i	C,B			

/soundbank					
/noHeight	B	R,C,	/set/heights	,i	1/0
			/set/evolution	,i	1/0
			/set/direction/reverse	,i	1/0
/overhead					
			/set/scale	,s	Scale index
/noCenterX	B	R			
TONALITY RECEIVES from TM on port 6160					
			/player/[id]/activity/discrete/hand/left		None
			/player/[id]/activity/discrete/hand/right		None
			/player/[id]/activity/discrete/head		None
			/player/[id]/activity/discrete/leg/left		None
			/player/[id]/activity/discrete/leg/right		None
			/player/[id]/activity/ discrete/body/upper		None
			/player/[id]/activity/discrete/body/lower		None
			/player/[id]/activity/discrete/body/right		None
			/player/[id]/activity/discrete/body/left		None
/activityNormal1	f	R	/player/[id]/activity/normal/hand/left	,f	Norm.
/activityNormal2	f	C,B	/player/[id]/activity/normal/hand/right	,f	Norm.
			/player/[id]/activity/normal/head	,f	Norm.
			/player/[id]/activity/normal/leg/left	,f	Norm.
			/player/[id]/activity/normal/leg/right	,f	Norm.
			/player/[id]/activity/normal/body/upper	,f	Norm.
			/player/[id]/activity/normal/body/lower	,f	Norm.
			/player/[id]/activity/normal/body/right	,f	Norm.
			/player/[id]/activity/normal/body/left	,f	Norm.
			/player/[id]/activity/peak	,f	Norm.
			/player/[id]/flow/leftwards/left		None
			/player/[id]/flow/leftwards/right		None
			/player/[id]/flow/rightwards/left		None
			/player/[id]/flow/rightwards/right		None
			/player/[id]/flow/upwards/left		None
			/player/[id]/flow/upwards/right		None
			/player/[id]/flow/downwards/left		None
			/player/[id]/flow/downwards/right		None
			/player/[id]/location/ready	,i	0/1
			/player/[id]/location/present	,i	0/1
			/player/[id]/location/centerX	,f	Norm.
			/player/[id]/location/centerZ	,f	Norm.
			/player/[id]/location/outOfRange		None
/height1	f	R	/player/[id]/position/height	,f	Norm.
/heightLevel1	i	R	/player/[id]/position/heightLevel	,i	0,1,2,3
			/player/[id]/position/vertical/hand/left	,f	Norm.
			/player/[id]/position/vertical/hand/right	,f	Norm.

			/player/[id]/position/side/hand/left	,f	Norm.
			/player/[id]/position/side/hand/right	,f	Norm.
			/player/[id]/position/side/foot/left	,f	Norm.
			/player/[id]/position/side/foot/right	,f	Norm.
			/player/[id]/position/front/hand/left	,f	Norm.
			/player/[id]/position/front/hand/right	,f	Norm.
			/player/[id]/position/front/foot/left	,f	Norm.
			/player/[id]/position/front/foot/right	,f	Norm.
/width1	f	R,C	/player/[id]/position/width	,f	Norm.
			/player/[id]/gesture/hit/overhead		None
/armSideLeft	f	C	/player/[id]/gesture/hit/side/left		None
/armSideRight	f	C	/player/[id]/gesture/hit/side/right		None
			/player/[id]/gesture/hit/down/left		None
			/player/[id]/gesture/hit/down/right		None
			/player/[id]/gesture/hit/forward/left		None
			/player/[id]/gesture/hit/ forward /right		None
			/player/[id]/gesture/kick/side/left		None
			/player/[id]/gesture/kick/side/right		None
			/player/[id]/gesture/kick/forward/left		None
			/player/[id]/gesture/kick/forward/right		None
			/player/[id]/gesture/doubleArmSide		None
			/player/[id]/gesture/doubleArmSideClose		None
			/player/[id]/gesture/jump		None
			/zone/[id]/activity/discrete		None
			/zone/[id]/activity/normal	,f	Norm.
			/zone/[id]/activity/flow/leftwards		None
			/zone/[id]/activity/flow/rightwards		None
			/zone/[id]/activity/flow/upwards		None
			/zone/[id]/activity/flow/downwards		None
			/zone/[id]/activity/flow/forward		None

GUI Elements

INTERFACE ELEMENTS

General controls	Player controls	Musical controls
Start (button)	1 Player / 2 Players / Zones (radio button)	Evolves Over Time (check box)
Stop (button)		Melody (dropdown list)
Volume (slider)		Scale (dropdown list)
Sensitivity (slider)	For each player : In-Space / In-Place (radio button)	Height (checkbox)
	For each player or zone: Instrument 1 (dropdown list) Instrument 2 (dropdown list) (this needs to be studied – its different for in-place as in-space...)	Chords (accents) (accents use different instruments)
		For in-space Direction reverse (checkbox)

Drums

Note to Composers

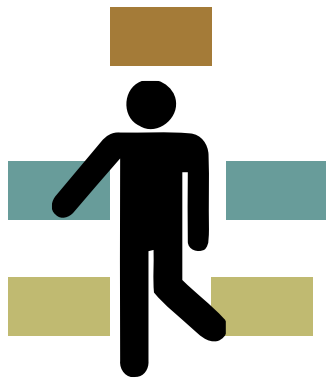
- We made a separate document for the music tasks of each ME. Look in the folder “MEWork” and you will see your tasks, and other information, up-to-date.

Introduction

You play a drum set. People with Down’s syndrome tend to love it because it is so intuitive and immediate. When you play single drum hits, it has high causality.

Description of Player Control Features

This diagram basically says it. There are 5 controllers:



After a certain number of notes are played (within 7 seconds), the notes become quantized. Also, if desired, after a certain number of notes are played, additional musical tracks come in.

The use of the arms is being developed so that any strong movement will work, whether it is:

- Quickly outward to the side
- Quickly forwards
- Downwards, like the motion of playing a drum (this is in development)

Kick side, will be combined with kick front. (so both will do the same thing)

Alphabet Used (marked in red)

				Drums Modes		
				1- (or 2-) Persons In-place	1-zone	2-zones
					I am not sure it has any value (let’s talk about it).	this is the normal techno ‘bed’.
/activity	/discrete	/hand	/left			
			/right			
		/head				
		/leg	/left			
			/right			
		/body	/upper			
			/lower			
			/right			
			/left			

	/normal	/hand	/left			
			/right			
		/head				
		/leg	/left			
			/right			
		/body	/upper			
			/lower			
			/right			
			/left			
	/peak					
	/flow	/leftwards	/left			
			/right			
		/rightwards	/left			
			/right			
		/upwards	/left			
			/right			
		/downwards	/left			
			/right			
/location	/ready			maybe before person is ready, no sound?		
	/present					
	/centerX					
	/centerZ					
	/outOfRange					
/position	/height					
	/heightLevel					
	/vertical	/hand	/left			
			/right			
	/side	/hand	/left			
			/right			
		/foot	/left			
			/right			
	/front	/hand	/left			
			/right			
		/foot	/left			
			/right			
	/width					
/gesture	/hit	/overhead		note5 (un-quantized)		
		/side	/left	note 1		
			/right	note 2		
		/down	/left	note 1		
			/right	note 2		
		/forward	/left	note 1		
			/right	note 2		
		/kick	/side	/left	note 3	
			/right	note 4		
			/forward	/left	note 3	
			/right	note 4		
	/doubleArmSide					
	/doubleArmSideClose					
	/jump					
	/clap					
/activity	/normal					strategy TBA (maybe a peak is useful)
	/discrete					strategy TBA

	/flow	/leftwards				
		/rightwards				
		/upwards				
		/downwards				

List of Messages Compared in MC 2.0 vs 3.0

MC 2.0			MC 3.0		
Message Pattern	Typetag	Mode (R,C,B)	Message Pattern	Typetag	Arguments
DRUMS SENDs on ports 9988(R,B) / 9989(C)			DRUMS SENDs to CM on port 6065		
			/set/loaded		None
			/set/player/[id]/soundbank/[#]/list /set/zone/[id]/soundbank/[#]/list	,s	String(s)
/ready	1	R,C	/set/ready		None
			DRUMS SENDs to TM on port 6061		
			/set/alphabet/[pattern] examples*: /set/alphabet/player/activity/discrete /set/alphabet/player/[id] /set/alphabet/player/[id]/activity /set/alphabet/zone/[id]/activity *: see OSC messages section for the rules	,i	1/0
DRUMS RECEIVES on ports 7032(R) / 7036(C) / 7886(B)			DRUMS RECEIVES from CM on port 6560		
/start	i	R,C	/set/play	,i	1
/stop	i	R,C			
/volume	,f	R,C	/set/volume	,f	volume
/sensitivity	f	R,C	/set/sensitivity	,f	sensitivity
			/set/player/[id]/soundbank/[#]	,i	Instrument's index
			/set/background	,s	always, newer, auto
			/set/evolution	,i	1/0
/patch	B	R,C			
			DRUMS RECEIVES from TM on port 6160		
			/player/[id]/activity/discrete/hand/left		None
			/player/[id]/activity/discrete/hand/right		None
			/player/[id]/activity/discrete/head		None

			/player/[id]/activity/discrete/leg/left		None
			/player/[id]/activity/discrete/leg/right		None
			/player/[id]/activity/ discrete/body/upper		None
			/player/[id]/activity/discrete/body/lower		None
			/player/[id]/activity/discrete/body/right		None
			/player/[id]/activity/discrete/body/left		None
			/player/[id]/activity/normal/hand/left	,f	Norm.
			/player/[id]/activity/normal/hand/right	,f	Norm.
			/player/[id]/activity/normal/head	,f	Norm.
			/player/[id]/activity/normal/leg/left	,f	Norm.
			/player/[id]/activity/normal/leg/right	,f	Norm.
			/player/[id]/activity/normal/body/upper	,f	Norm.
			/player/[id]/activity/normal/body/lower	,f	Norm.
			/player/[id]/activity/normal/body/right	,f	Norm.
			/player/[id]/activity/normal/body/left	,f	Norm.
			/player/[id]/activity/peak	,f	Norm.
			/player/[id]/flow/leftwards/left		None
			/player/[id]/flow/leftwards/right		None
			/player/[id]/flow/rightwards/left		None
			/player/[id]/flow/rightwards/right		None
			/player/[id]/flow/upwards/left		None
			/player/[id]/flow/upwards/right		None
			/player/[id]/flow/downwards/left		None
			/player/[id]/flow/downwards/right		None
			/player/[id]/location/ready	,i	0/1
			/player/[id]/location/present	,i	0/1
			/player/[id]/location/centerX	,f	Norm.
			/player/[id]/location/centerZ	,f	Norm.
			/player/[id]/location/outOfRange		None
			/player/[id]/position/height	,f	Norm.
			/player/[id]/position/heightLevel	,i	0,1,2,3
			/player/[id]/position/vertical/hand/left	,f	Norm.
			/player/[id]/position/vertical/hand/right	,f	Norm.
			/player/[id]/position/side/hand/left	,f	Norm.
			/player/[id]/position/side/hand/right	,f	Norm.
			/player/[id]/position/side/foot/left	,f	Norm.
			/player/[id]/position/side/foot/right	,f	Norm.
			/player/[id]/position/front/hand/left	,f	Norm.
			/player/[id]/position/front/hand/right	,f	Norm.
			/player/[id]/position/front/foot/left	,f	Norm.
			/player/[id]/position/front/foot/right	,f	Norm.
			/player/[id]/position/width	,f	Norm.
/overhead	B	R,C	/player/[id]/gesture/hit/overhead		None
/SideHandL1	B	R,C	/player/[id]/gesture/hit/side/left		None
/SideHandR1	B	R,C	/player/[id]/gesture/hit/side/right		None
			/player/[id]/gesture/hit/down/left		None

			/player/[id]/gesture/hit/down/right		None
			/player/[id]/gesture/hit/forward/left		None
			/player/[id]/gesture/hit/ forward /right		None
/SideFootL1	B	C	/player/[id]/gesture/kick/side/left		None
/SideFootR1	B	C	/player/[id]/gesture/kick/side/right		None
			/player/[id]/gesture/kick/forward/left		None
			/player/[id]/gesture/kick/forward/right		None
			/player/[id]/gesture/doubleArmSide		None
			/player/[id]/gesture/doubleArmSideClose		None
/jump	B	R,C	/player/[id]/gesture/jump		None
			/zone/[id]/activity/discrete		None
			/zone/[id]/activity/normal	,f	Norm.
			/zone/[id]/activity/flow/leftwards		None
			/zone/[id]/activity/flow/rightwards		None
			/zone/[id]/activity/flow/upwards		None
			/zone/[id]/activity/flow/downwards		None
			/zone/[id]/activity/flow/forward		None

GUI

INTERFACE ELEMENTS

General controls	Player controls	Musical controls
Start (button)	1 Player / 2 Players / Zones (radio button)	Evolves Over Time (check box)
Stop (button)	For each player or zone: drum set)	Background Music (radio button): <ul style="list-style-type: none"> Always Never Auto
Volume (slider)		
Sensitivity (slider)		

Particles

Note to Composers

- We made a separate document for the music tasks of each ME. Look in the folder “MEWork” and you will see your tasks, and other information, up-to-date.

Alphabet used (marked in red)	Particles Modes						
				1-Pers. (or2-pers.) In-Space	1-Pers. (or2-pers.) In-Place	1-zone	2-zones
						I'm not sure we need this option (let's talk about it).	
/activity	/discrete	/hand	/left	plays single particles	plays single particles		

			/right		plays single particles			
		/head			plays single particles			
		/leg	/left					
			/right					
		/body	/upper					
			/lower					
			/right			plays single particles		
			/left			plays single particles		
	/normal	/hand	/left	plays music	plays music			
			/right		plays music			
		/head			plays music (could be random L/R, or something special)			
		/leg	/left					
			/right					
		/body	/upper					
			/lower					
			/right					
	/left							
	/peak							
	/flow	/leftwards	/left					
			/right					
		/rightwards	/left					
			/right					
		/upwards	/left					
			/right					
		/downwards	/left					
			/right					
	/location	/ready				delay playing until player is ready (we need to test this)		
		/present				maybe it plays, but quietly, before ready.		
		/centerX						
/centerZ								
/outOfRange								
/position	/height			in hL1 and 0 – melts sounds	chooses particles when no arms are found*			
	/heightLevel			hL0 – could be something special (using activity and height to modulate) hL3 – could be something special?	This is new: hL0 – something special? (using activity and height to modulate)			
	/vertical	/hand	/left		chooses particles when arms are found*			
			/right					
	/side	/hand	/left					
			/right					

		/foot	/left				
			/right				
	/front	/hand	/left				
			/right				
		/foot	/left				
			/right				
	/width			shapes longs	shapes longs?		
/gesture	/hit	/overhead		triggers longs	could trigger longs?		
		/side	/left				
			/right				
		/down	/left				
			/right				
	/forward	/left	/left				
			/right				
		/right	/left				
			/right				
	/kick	/side	/left				
			/right				
		/forward	/left	maybe??	maybe??		
			/right	maybe??	maybe??		
	/doubleArmSide			kills longs	kills longs?		
	/doubleArmSideClose						
	/jump						
	/clap						
/activity	/normal					plays music	strategy for 2zones ?
	/discrete					plays discrettes	plays discrettes
	/flow	/leftwards					
		/rightwards					
		/upwards					
		/downwards					

* - this needs to be discussed.

List of Messages Compared in MC 2.0 vs 3.0

MC 2.0			MC 3.0		
Message Pattern	Typetag	Mode (R,C,B)	Message Pattern	Typetag	Arguments
PARTICLES SENDs on ports 9988(R,B) / 9989(C)			PARTICLES SENDs to CM on port 6065		
			/set/loaded		None
			/set/player/[id]/soundbank/[#]/list /set/zone/[id]/soundbank/[#]/list	,s	String(s)
/ready	1	R,C,B	/set/ready		None
			PARTICLES SENDs to TM on port 6061		
			/set/alphabet/[pattern]	,i	1/0
			examples*: /set/alphabet/player/activity/discrete /set/alphabet/player/[id] /set/alphabet/player/[id]/activity		

			/set/alphabet/zone/[id]/activity		
			*: see OSC messages section for the rules		
PARTICLES RECEIVES on ports 7032(R) / 7036(C) / 7886(B)			PARTICLES RECEIVES from CM on port 6560		
/start	i	R,C,B	/set/play	,i	1
/stop	i	R,C,B			
/volume	,f	R,C,B	/set/volume	,f	volume
			/set/sensitivity	,f	sensitivity
/soundbank	i	R,C,B	/set/player/[id]/soundbank/[#]	,i	Instrument's index
/noHeight,	B	R,C	/set/noHeights	,i	1/0
/killLong	B	R.C	/set/killLongs	,i	1/0
			/set/rthym	,i	1/0
			PARTICLES RECEIVES from TM on port 6160		
			/player/[id]/activity/discrete/hand/left		None
			/player/[id]/activity/discrete/hand/right		None
/activitySensitive	B	R	/player/[id]/activity/discrete/head		None
			/player/[id]/activity/discrete/leg/left		None
			/player/[id]/activity/discrete/leg/right		None
			/player/[id]/activity/ discrete/body/upper		None
			/player/[id]/activity/discrete/body/lower		None
/activitySensitive1	B	C	/player/[id]/activity/discrete/body/right		None
/activitySensitive2	B	C	/player/[id]/activity/discrete/body/left		None
/armSideLeft	f	C	/player/[id]/activity/normal/hand/left	,f	Norm.
/armSideRight	f	C	/player/[id]/activity/normal/hand/right	,f	Norm.
			/player/[id]/activity/normal/head	,f	Norm.
			/player/[id]/activity/normal/leg/left	,f	Norm.
			/player/[id]/activity/normal/leg/right	,f	Norm.
			/player/[id]/activity/normal/body/upper	,f	Norm.
			/player/[id]/activity/normal/body/lower	,f	Norm.
/activityNormal1	B	R,C,B	/player/[id]/activity/normal/body/right	,f	Norm.
/activityNormal2		C,B	/player/[id]/activity/normal/body/left	,f	Norm.
			/player/[id]/activity/peak	,f	Norm.
			/player/[id]/flow/leftwards/left		None
			/player/[id]/flow/leftwards/right		None
			/player/[id]/flow/rightwards/left		None
			/player/[id]/flow/rightwards/right		None
			/player/[id]/flow/upwards/left		None
			/player/[id]/flow/upwards/right		None
			/player/[id]/flow/downwards/left		None
			/player/[id]/flow/downwards/right		None
			/player/[id]/location/ready	,i	0/1
			/player/[id]/location/present	,i	0/1

/centerX	f	R,C	/player/[id]/location/centerX	,f	Norm.
			/player/[id]/location/centerZ	,f	Norm.
			/player/[id]/location/outOfRange		None
/height1	f	R,C	/player/[id]/position/height	,f	Norm.
/heightLevel1	i	R	/player/[id]/position/heightLevel	,i	0,1,2,3
			/player/[id]/position/vertical/hand/left	,f	Norm.
			/player/[id]/position/vertical/hand/right	,f	Norm.
			/player/[id]/position/side/hand/left	,f	Norm.
			/player/[id]/position/side/hand/right	,f	Norm.
			/player/[id]/position/side/foot/left	,f	Norm.
			/player/[id]/position/side/foot/right	,f	Norm.
			/player/[id]/position/front/hand/left	,f	Norm.
			/player/[id]/position/front/hand/right	,f	Norm.
			/player/[id]/position/front/foot/left	,f	Norm.
			/player/[id]/position/front/foot/right	,f	Norm.
/width1	f	R,C	/player/[id]/position/width	,f	Norm.
/overhead1	B	R,C	/player/[id]/gesture/hit/overhead		None
			/player/[id]/gesture/hit/side/left		None
			/player/[id]/gesture/hit/side/right		None
			/player/[id]/gesture/hit/down/left		None
			/player/[id]/gesture/hit/down/right		None
			/player/[id]/gesture/hit/forward/left		None
			/player/[id]/gesture/hit/ forward /right		None
			/player/[id]/gesture/kick/side/left		None
			/player/[id]/gesture/kick/side/right		None
			/player/[id]/gesture/kick/forward/left		None
			/player/[id]/gesture/kick/forward/right		None
			/player/[id]/gesture/doubleArmSide		None
			/player/[id]/gesture/doubleArmSideClose		None
			/player/[id]/gesture/jump		None
			/zone/[id]/activity/discrete		None
			/zone/[id]/activity/normal	,f	Norm.
			/zone/[id]/activity/flow/leftwards		None
			/zone/[id]/activity/flow/rightwards		None
			/zone/[id]/activity/flow/upwards		None
			/zone/[id]/activity/flow/downwards		None
			/zone/[id]/activity/flow/forward		None

GUI

General controls	Player controls	Musical controls
Start (button)	1 Player / 2 Players / Zones (radio button)	Evolving mode (check box)
Stop (button)	For each player or zone:	Long Active (check box)
Volume (slider)	Soundbank 1 (dropdown list)	Kill Long, trigger long (push

Sensitivity (slider)	Soundbank 2 (dropdown list)	button) Height Active (check box) Rhythms Active (check box) Accents Active (check box)
-----------------------	-----------------------------	--

Import-Your-Own-Music

Note to Composers

- We made a separate document for the music tasks of each ME. Look in the folder “MEWork” and you will see your tasks, and other information, up-to-date.

Importer (“Import-Your-Own-Music”, etc. we have not found a title yet)

DESCRIPTION

We have done “Import” many times. It is an important part of the MC1. There are therapists who are currently using it. Josepha has used it as well. So, in its basic form, it is not a new idea. Having said this, there are certain “extensions” we could imagine. These are complicated for a couple of reasons:

- 1) some have been tested. many have not.
- 2) they are difficult to imagine working universally – that is, just because they work for one piece of music, does not mean that they will work for EVERY piece of music.

Thus, we really have to think of IMPORTER in two phases:

Phase I: the basic version we have used in the past – based on 1 Zone.

Phase II: various extensions that we can test for possible future implication.

Phase I Implementation

USER EXPERIENCE

1. User has a favorite song they want to use. First they have to download, buy, generate, etc. an mp3 format file of the music they want to use.

It would be nice to be able to also allow better-quality formats, such as wav. and .aif. The problem, I can imagine, is that there are so many different formats – sampling rates, depth (bitrate), etc. – I do not know if you can accept everything.

2. The copy the file onto a USB stick, and stick it in the MC3.

3. A pop-up appears, asking: Click on the Music File you wish to use.

4. After making their choice, their music appears in the ME “available songs” drop-down menu.

5. They then use the ME with the following features:

- a) more movement = louder, brighter sound; less movement = quieter, more muffled sound (like hearing it from a distance, or from inside a box). (low-pass or high-pass filters, we will have to test)

- b) when the player stops moving, the music either stops, or becomes very quiet. (see GUI below)

c) there can be a pitch bend as the song “gets going”. So it sounds like a vinyl record starting up, and slowing down. Normally of course, by normal ranges of activity, there is no pitch bend. It only occurs when starting up from stillness, or dying down to stillness. (this will need a time-based filter, i.e. a stop shorter than 500mS does not cause the bend, etc.).

DATA STREAMS

Red indicates, streams needed.

PHASE I				1-zone
				this is the normal mode
/activity	/discrete	/hand	/left	
			/right	
		/head		
		/leg	/left	
			/right	
		/body	/upper	
			/lower	
			/right	
			/left	
	/normal	/hand	/left	
			/right	
		/head		
		/leg	/left	
			/right	
		/body	/upper	
			/lower	
			/right	
			/left	
	/peak			
	/flow	/leftwards	/left	
			/right	
		/rightwards	/left	
			/right	
		/upwards	/left	
			/right	
		/downwards	/left	
			/right	
/location	/ready			
	/present			
	/centerX			
	/centerZ			
	/outOfRange			
/position	/height			
	/heightLevel			
	/vertical	/hand	/left	
			/right	
	/side	/hand	/left	
			/right	
		/foot	/left	
			/right	
	/front	/hand	/left	
			/right	
		/foot	/left	
			/right	

	/width			
/gesture	/hit	/overhead		
		/side	/left	
			/right	
		/down	/left	
			/right	
		/forward	/left	
			/right	
	/kick	/side	/left	
			/right	
		/forward	/left	
			/right	
	/doubleArmSide			
	/doubleArmSideClose			
	/jump			
	/clap			
/activity	/normal			energy level, and start-stop procedure (described above)
	/discrete			
	/flow	/leftwards		we don't know if we are going to have this... if so, it could be interesting. (e.g. pitch is slightly higher in one direction than the other)
		/rightwards		
		/upwards		
		/downwards		

GUI Elements

INTERFACE ELEMENTS

General controls	Player controls	Musical controls
Start (button)	song (dropdown list)	(Radio Button)
Stop (button)		Music Stops When Player Stops (default) Music Never Stops
Volume (slider)		Reset (resets song to the beginning)
Sensitivity (slider)		(button)

Phase II Implementation

As I said above, there are number of questions which needs to be addressed before we consider this. Tests will be needed to know of if any of this makes sense.

Of course, Phase II could be a set of options that are available, but that are turned off by default.

Still, I don't want them in the MC unless they work in most cases, and this will need to be proven to me.

Phase II involves a lot of music composition issues, so we will need help with it.

Extended options in Phase II: (in addition to what happens in Phase I)

- accents
- discretes
- height stuff (melting down, and HL0)
- intelligent filtering

DATA STREAMS

Red indicates, streams needed.

PHASE II					1-Person
	/activity	/discrete	/hand	/left	(this could offer advantages over /zones/[id]/activity/discrete since therapist can be excluded. On the other side, these players are mostly in beds or chairs, so the body tracking may –in most cases – not work)
				/right	
			/head		
			/leg	/left	
				/right	
			/body	/upper	
				/lower	
				/right	
				/left	
		/normal	/hand	/left	(this could offer advantages over /zones/[id]/activity/normal since therapist can be excluded. On the other side, these players are mostly in beds or chairs, so the body tracking may –in most cases – not work)
				/right	
			/head		
			/leg	/left	
				/right	
			/body	/upper	
				/lower	
				/right	
				/left	
		/peak			biggest level sound. See Accents below.
		/flow	/leftwards	/left	
				/right	
			/rightwards	/left	
				/right	
			/upwards	/left	
				/right	
			/downwards	/left	
				/right	
	/location	/ready			??
		/present			??
		/centerX			
		/centerZ			
		/outOfRange			
	/position	/height			in hL1 and 0 –classic melt down

		/heightLevel			hL0 – is something special (granulation was used successfully in techno) hL3 – could be something special, such as a very bright high-pass filter.
		/vertical	/hand	/left	
				/right	
		/side	/hand	/left	
				/right	
			/foot	/left	
				/right	
		/front	/hand	/left	
				/right	
			/foot	/left	
				/right	
		/width			
	/gesture	/hit	/overhead		accent*
			/side	/left	accent*
				/right	accent*
			/down	/left	accent*
				/right	accent*
			/forward	/left	accent*
				/right	accent*
		/kick	/side	/left	accent*
				/right	accent*
			/forward	/left	accent*
				/right	accent*
		/doubleArmSide			accent*
		/doubleArmSideClose			
		/jump			accent*
		/clap			
/zones	/activity	/normal			energy level, and start-stop procedure
		/discrete			Small, interesting small sounds. a-tonal. (hear examples 'A')
		/flow	/leftwards		
			/rightwards		
			/upwards		
			/downwards		

ACCENTS

Accents are used in many of the environments to add fun and causality. There are six different kinds of accent gestures:

- peak
- kick
- jump
- overhead
- hits (side, front or down)
- das

But this doesn't mean there are six different sounds! It is usually better not to have too many different kinds of accents. Thus, many times, different gestures make the same sound.

Each accents is not a single sound, but a collection of similar samples, usually 7-10. Also, they are "naturalized", so that even if you hear the same exact sample twice, it will be a little different.

Musically, they present a challenge, especially here, since on the one hand, they have to contrast strongly with the other notes being played. But on the other hand, they have to fit musically! With tonal music, for example, this means they have to be in the same key as the rest of the song, otherwise, they can sound pretty “wrong”.

Some sounds, such as vocalizations (shouts especially), animal sounds, or some kinds of percussion, are atonal – and will work well with any key.

(The point obviously, is that we do not know the key of the song that the user is importing. There may be software to analyze both the key, and the pitch (frequency) of the imported song, so there may be some possibilities for “intelligent” accents. (Though I doubt it will work well. I refer you to Andreas. He is something of an expert. *

Finally, not every player is able to play every accent. Remember, “Importer” is primarily for severely disabled persons! This is why it is considered as an operator choice:

Accents:

☒ - all active

☐ - none active

☒ - peak

☒ - kick

☒ - jump

☒ - hand hit (side or front or down*)

☒ - overhead hit

☒ - double arm side

* - He suggested once that we might be able to pull out the base, or drums out from a song. There is software for this. HOWEVER, personally,,,, I am skeptical. I doubt it works very well, or very universally. But if it did, this could offer a more sophisticated role for the player – more like in Techno, where they get to bring in the base and drums and with their bodies.

GUI Elements

INTERFACE ELEMENTS

General controls	Player controls	Musical controls
Start (button)	song (dropdown list)	(Radio Button)
Stop (button)		Music Stops When Player Stops (default) Music Never Stops
Volume (slider)		Reset (resets song to the beginning) (button)
Sensitivity (slider)		Discretes active
		Accents active ... see above

List of Messages Compared in MC 2.0 vs 3.0

MC 2.0			MC 3.0		
Message Pattern	Typetag	Mode (R,C,B)	Message Pattern	Typetag	Arguments
IMPORTER SENDs on ports 9988(R,B) / 9989(C)			IMPORTER SENDs to CM on port 6065		
			/set/loaded		None
			/set/player/[id]/soundbank/[#]/list /set/zone/[id]/soundbank/[#]/list	,s	String(s)
/ready	1	R,C,B	/set/ready		None
			IMPORTER SENDs to TM on port 6061		
			/set/alphabet/[pattern] examples*: /set/alphabet/player/activity/discrete /set/alphabet/player/[id] /set/alphabet/player/[id]/activity /set/alphabet/zone/[id]/activity *: see OSC messages section for more	,i	1/0
IMPORTER RECEIVES on ports 7032(R) / 7036(C) / 7886(B)			IMPORTER RECEIVES from CM on port 6560		
/start	i	R,C,B	/set/play	,i	1
/stop	i	R,C,B			
/volume	,f	R,C,B	/set/volume	,f	volume
			/set/sensitivity	,f	sensitivity
			IMPORTER RECEIVES from TM on port 6160		
			/player/[id]/activity/discrete/hand/left		None
			/player/[id]/activity/discrete/hand/right		None
			/player/[id]/activity/discrete/head		None
			/player/[id]/activity/discrete/leg/left		None
			/player/[id]/activity/discrete/leg/right		None
			/player/[id]/activity/discrete/body/upper		None
			/player/[id]/activity/discrete/body/lower		None
			/player/[id]/activity/discrete/body/right		None
			/player/[id]/activity/discrete/body/left		None
			/player/[id]/activity/normal/hand/left	,f	Norm.

			/player/[id]/activity/normal/hand/right	,f	Norm.
			/player/[id]/activity/normal/head	,f	Norm.
			/player/[id]/activity/normal/leg/left	,f	Norm.
			/player/[id]/activity/normal/leg/right	,f	Norm.
			/player/[id]/activity/normal/body/upper	,f	Norm.
			/player/[id]/activity/normal/body/lower	,f	Norm.
			/player/[id]/activity/normal/body/right	,f	Norm.
			/player/[id]/activity/normal/body/left	,f	Norm.
			/player/[id]/activity/peak	,f	Norm.
			/player/[id]/flow/leftwards/left		None
			/player/[id]/flow/leftwards/right		None
			/player/[id]/flow/rightwards/left		None
			/player/[id]/flow/rightwards/right		None
			/player/[id]/flow/upwards/left		None
			/player/[id]/flow/upwards/right		None
			/player/[id]/flow/downwards/left		None
			/player/[id]/flow/downwards/right		None
			/player/[id]/location/ready	,i	0/1
			/player/[id]/location/present	,i	0/1
			/player/[id]/location/centerX	,f	Norm.
			/player/[id]/location/centerZ	,f	Norm.
			/player/[id]/location/outOfRange		None
			/player/[id]/position/height	,f	Norm.
			/player/[id]/position/heightLevel	,i	0,1,2,3
			/player/[id]/position/vertical/hand/left	,f	Norm.
			/player/[id]/position/vertical/hand/right	,f	Norm.
			/player/[id]/position/side/hand/left	,f	Norm.
			/player/[id]/position/side/hand/right	,f	Norm.
			/player/[id]/position/side/foot/left	,f	Norm.
			/player/[id]/position/side/foot/right	,f	Norm.
			/player/[id]/position/front/hand/left	,f	Norm.
			/player/[id]/position/front/hand/right	,f	Norm.
			/player/[id]/position/front/foot/left	,f	Norm.
			/player/[id]/position/front/foot/right	,f	Norm.
			/player/[id]/position/width	,f	Norm.
			/player/[id]/gesture/hit/overhead		None
			/player/[id]/gesture/hit/side/left		None
			/player/[id]/gesture/hit/side/right		None
			/player/[id]/gesture/hit/down/left		None
			/player/[id]/gesture/hit/down/right		None
			/player/[id]/gesture/hit/forward/left		None
			/player/[id]/gesture/hit/ forward /right		None
			/player/[id]/gesture/kick/side/left		None
			/player/[id]/gesture/kick/side/right		None
			/player/[id]/gesture/kick/forward/left		None
			/player/[id]/gesture/kick/forward/right		None

			/player/[id]/gesture/doubleArmSide		None
			/player/[id]/gesture/doubleArmSideClose		None
			/player/[id]/gesture/jump		None
			/zone/[id]/activity/discrete		None
			/zone/[id]/activity/normal	,f	Norm.
			/zone/[id]/activity/flow/leftwards		None
			/zone/[id]/activity/flow/rightwards		None
			/zone/[id]/activity/flow/upwards		None
			/zone/[id]/activity/flow/downwards		None
			/zone/[id]/activity/flow/forward		None

ADDITIONAL PRODUCT POSSIBILITIES

Description

The device described above, the MC3.0, is the primary configuration of the MotionComposer. It is designed for sale to therapists, special schools, children's schools, centers for persons with disabilities.

there are, however, other possible markets:

- Developers, Game Developers
- Performers
- Artists – creators of installations, choreographers, composers, performance artists, opera, etc.
- Musicians
- Music Pedagogues
- Researchers

For such users, the MC3.0 is much too simple, and closed. All the 3.0 can do is play fixed, limited, music. This led us to consider 2 other possible products. These will be released after the MC3.0.

MC PRO-VERSION

With A much more powerful software platform, it can be used for many purposes. Of course, it is not so easy-to-use. It essentially makes available our motion tracking and mapping expertise, for other uses (movement can be used to control lights, external music devices, video projections, etc. There are endless possibilities.

Our plan is to build e talked about a simple MAX/msp -style interface, offering the OSC/UDP data via Ethernet. Thus, the "Pro-Version" is just like the MC3.0, but without the music.

Another possibility is to sell the same chassis but without the motherboard inside. The user would connect the box to their own computer via Ethernet cable. (Though this sounds tricky for technical reasons).

MC MUSICIANS' VERSION

Again, the difference is in the software.

This began as the so-called Tonality2, which was written by Ives for our possible collaboration with Stevie Wonder. It allows programmable chords to be made available to be triggered in much the way Drums is played.

Judging from our experience at Orff Institute – where we presented MC to at music pedagogy conference, this could be a thing.

Ives has other ideas to make this version useful for music teaching.