MULTITACTION

MT CELL USER MANUAL

MT553UTB

MT555UTB

Revision 6
July 2017

Safety Information

Read through carefully and understand all the safety information before installing or using the product. Follow the instructions in this manual and marked on the MultiTaction Cell. Take careful notice of all warnings and safety-related instructions to install and use the product safely.

Danger: *Indicates a hazard with a high level of risk which, if not avoided, will result in death or serious injury AND / OR property damage.*

Warning: *Indicates a hazard with a medium level of risk which, if not avoided, could result in death or serious injury AND / OR property damage.*

Caution: *Indicates a hazard with a low level of risk which, if not avoided, could result in minor or moderate injury AND / OR property damage.*

Note: *Indicates important information.*

Consignes de sécurité

Etudier soigneusement et bien comprendre toutes les consignes de sécurité avant d'installer ou d'utiliser le produit. Respecter les instructions fournies dans ce manuel et celles marquées sur la MultiTaction Cell. Bien tenir compte de tous les avertissements et des instructions de sécurité afin d'installer et d'utiliser le produit en toute sécurité.

Danger: *Indique un danger avec un niveau de risque élevé qui, s'il n'est pas évité, peut causer la mort ou une blessure grave ET / OU des dommages matériels.*

Avertissement: *Indique un danger avec un niveau de risque élevé qui, s'il n'est pas évité, peut causer la mort ou une blessure grave ET / OU des dommages matériels.*

Attention: *Indique un danger avec un niveau de risque bas qui, s'il n'est pas évité, peut causer des blessures mineures ou légères ET / OU des dommages matériels.*

Remarque: *Contient des informations importantes.*

Regulatory information

European Union Directive on Restriction of Hazardous Substances (RoHS)

This MultiTaction Cell complies with the following regulatory standards:

- RoHS
- EN 55022
- EN 55024
- IEC 61000-3-2
- IEC 61000-3-3
- IEC 62471
- IEC60950-1:2005+A1:2009

For more compliance information, please provide MultiTaction with the serial number of the MultiTaction Cell.

Disposal Guidelines

The MultiTaction Cell complies with the European RoHS -directive. Please consult local authorities and professionals when planning to dispose of the Cell.

Contents

1 Introduction

MultiTaction Cells (or *MT Cells*) are multi-touch, modular LCD displays intended for, but not limited to, retail businesses, advertising, exhibitions, museums, education, and design installations.

'MultiTaction' means that Cells can track and react to several people interacting with them simultaneously. Moreover, the system tracks a person's hands instead of points of contact only, further enhancing interactive possibilities. 'Modular' means that Cells can be easily stacked and combined to form a video wall (a single large display array), showing interactive content and reacting to multiple users' touch.

The LCD displays deliver full HD picture quality, have a long life-span, and need hardly any maintenance. Current models also have an Ultra-Thin bezel (UTB).

Cells need an external computer for running the applications. You can work on Linux, OS X, or Windows environments, and create your own tailored apps through the MultiTouch Cornerstone SDK or many other supported methods.

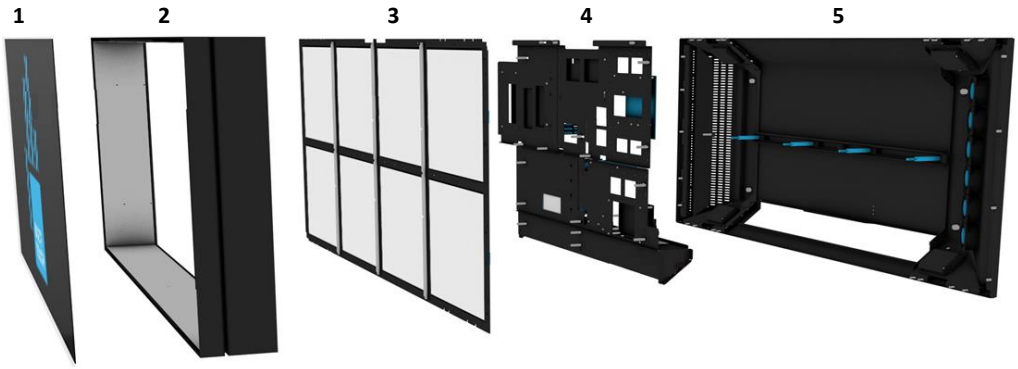
This guide contains installation and operating instructions for MultiTaction 55" ultra-thin bezel Cells. Current models are the MT553UTB and MT555UTB.

MultiTaction Cell

2 Cell overview

Note: *For full technical specifications, see Appendix A.*

MultiTaction is an innovative way to build an integrated multi-touch LCD display. The front glass and LCD package is tuned to allow the integrated optical imaging system to work effectively through it. Imaging logic and processing power are embedded in the display unit. The diagram below shows the main components used in MultiTaction Cells.



MultiTaction Cell technologies

1 *LCD panel. Uses Computer Vision Through Screen (CVTS) touch technology.*

2 *Aluminium frame*

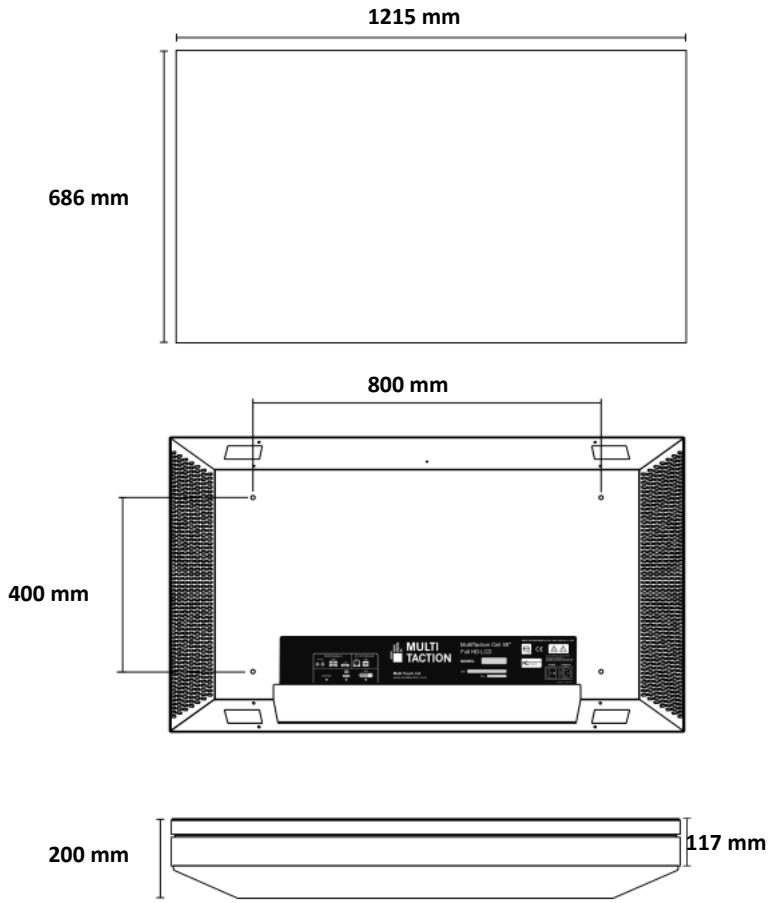
3 *Backlight. Incorporates an Integrated Backlight Emitter Camera (IBEC) module*

4 *Electronics, including:*
- *Extensible Hybrid Tracking Engine (EHTE) ie, the core tracking technology*
- *Matrix Tracking System (MTS), comprising 32 distributed cameras*
- *Multi Format Tracking Output (MFTO), providing compatibility with most touch protocols*

5 *Cooling backplate*

2.1 Dimensions and 3D models

A Cell has the following outer dimensions:



Cell dimensions: MT553UTB and MT555UTB

3D models of the MT553UTB and MT555UTB Cells are available on the Download page of the MultiTaction website:

https://cornerstone.multitouch.fi/download

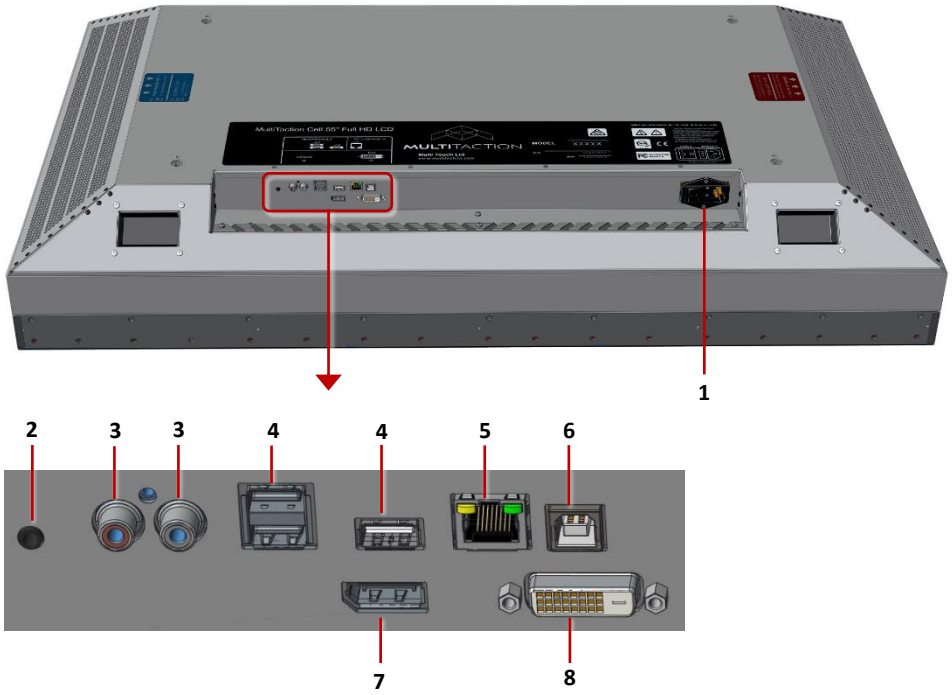
The following zip files are available to download:

- MT555UTB_CUSTOMER_MODELS.zip includes Cell 3D models in various formats.
- MT555UTB_CUSTOMER_MODELS.zip includes Cell 3D models in various formats.
- MT553UTB_3D_ACIS-FILE.zip includes a Cell 3D model in ACIS SAT format.

Note that you will need to register before you can download the 3D models.

2.2 Connection panel

The Cell connection panel is located at the back of the device. It contains the power switch and sockets for connecting the Cell to: your network or an external application computer; a power supply; and a local mouse and keyboard.



Cell connection panel. Includes the following sockets

1	*Mains*	*Power connection to the mains supply. Mains power switch. Two fuses*
2	*3.5mm socket*	*Not used*
3	*RCA*	*Not used*
4	*USB 2.0 Type A*	*Three ports for peripheral devices, especially keyboard and mouse*
5	*Ethernet*	*For control input*
6	*USB Type B*	*Not used*
7	*DisplayPort*	*Not used*
8	*DVI-D*	*Digital signal input*

2.3 Warning labels

The following warning labels can be found on or inside the Cell:

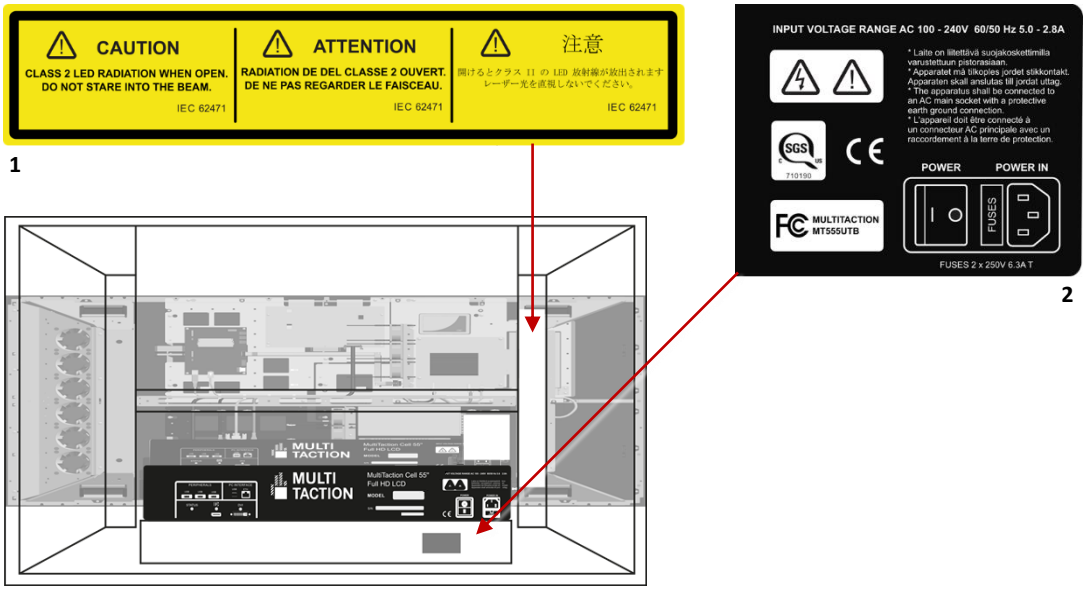
- CAUTION. CLASS 2 LED RADIATION WHEN OPEN. DO NOT STARE INTO THE BEAM.

  ATTENTION. RAYONNEMENT LED DE CATÉGORIE 2 UNE FOIS OUVERT. NE PAS REGARDER DANS LE FAISCEAU.

- Product label
  - Laite on liitettävä suojakoskettimilla varustettuun pistorasiaan.
  - Apparatet må tilkoples jordet stikkontakt. Apparaten skall anslutas till jordat uttag.
  - The apparatus shall be connected to an AC main socket with a protective earth ground connection.
  - L'appareil doit être connecté à un connecteur AC principale avec un raccordement à la terre de protection.
- CAUTION. CLASS 2 LED RADIATION WARNING LABEL (IEC 62471)

  ATTENTION. RAYONNEMENT LED DE CATÉGORIE 2 (CEI 62471)

  Attached to metal structure inside the Cell enclosure. Not shown in diagrams below.



*Warning labels on rear of Cell: **1** CLASS 2 LED RADIATION WHEN OPEN. **2** Product label.*

3 Unpacking the Cell

The Cell is delivered in a specially designed cardboard box to secure it during transportation.

You need two people to remove the Cell from its box, one person lifting from each side. Make sure you have enough space around the cardboard box to lift the Cell and position it in the desired location.

3.1 What's in the box?

MultiTaction Cell product box contains the following items:

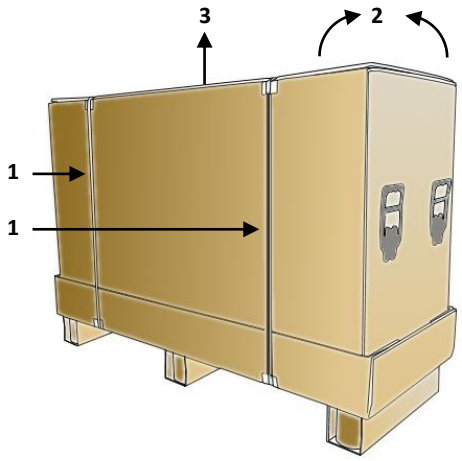
- MultiTaction Cell
- A pair of desk stands
- Protective/auto-calibration sheet
- English user manual

Carefully check that you have received all the items before proceeding to install the Cell. If your product box is missing an item, please contact your dealer.

3.2 Unpack the cardboard box

To unpack the Cell from its cardboard shipping box, place the shipping box on a clean floor. Then:

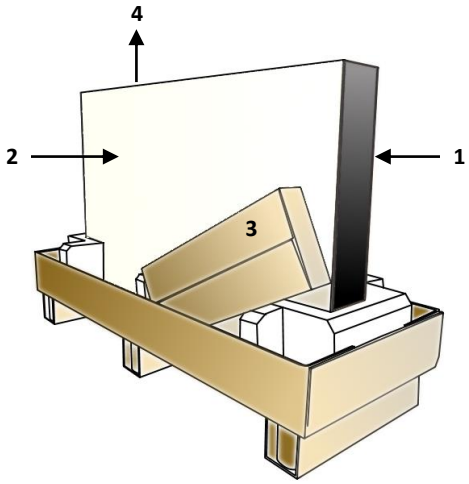
1. Cut the packing straps.
2. Open the top lid of the box.
3. Remove the protective Cell padding. Then remove the top part of the box.



1 *Packing straps.* **2** *Open the lid.* **3** *Remove the padding and the top part of the box.*

4. Remove the Cell from the container with help from another person, one lifting from each side.
5. Retain the white cardboard *calibration sheet*. You will need the calibration sheet to calibrate the Cell in section 8.4.1.

   Important! *Retain at least one of your Cell calibration sheets after setting up your video wall! Do not throw them all away. There may be occasions in the future when you need to recalibrate individual Cells.*



1 Cell. 2 Calibration sheet. 3 Accessories box. 4 Remove the Cell.

4 Before you start

4.1 Proper environment

Choose a well-ventilated, cool space for setting up the display.

Caution

- *Radiating heat sources (like halogen spots) can damage the display.*
- *Keep the Cell away from strong spot-light areas.*
- *Intense sunlight can damage the display.*
- *Place the Cell in a space where intense sunlight does not hit the display area (front panel).*

Avertissement

- *Les sources de dégagement de chaleur rayonnée (notamment les lampes halogènes) peuvent endommager l'écran.*
- *Maintenir la Cellule éloignée des zones à forte source de lumière.*
- *Une lumière solaire intense peut endommager l'affichage.*
- *Placer la cellule dans un endroit où les rayons solaires intenses n'atteignent pas l'affichage (panneau avant).*

4.2 Normal operating conditions

MultiTaction Cells are designed to operate normally for extended periods of time (up to 20 hours per day). Using a Cell under normal operating conditions ensures maximum performance.

The following table contains the normal operating conditions for Cells.

Condition	Correct parameters
Temperature	0° to 35° C (32° to 95° F). Optimal performance 20° to 25° C (68° to 77° F)
Humidity	Non-condensing < 80% RH
Display pattern	A moving picture or regularly changing animation
Sunlight	No intense sunlight on the display area (front panel)
Infrared sources	Do not use in the vicinity of strong spot-light areas or other strong infrared sources
Radiating heat	Do not use in the vicinity of strong radiating heat sources
Dust	Clean dust-free environment

Caution

- *Using the Cell under other than the Normal Operating Conditions may reduce the lifetime of the Cell.*
- *To reach the lifetime specification of the Cell always use it under the Normal Operating Conditions.*

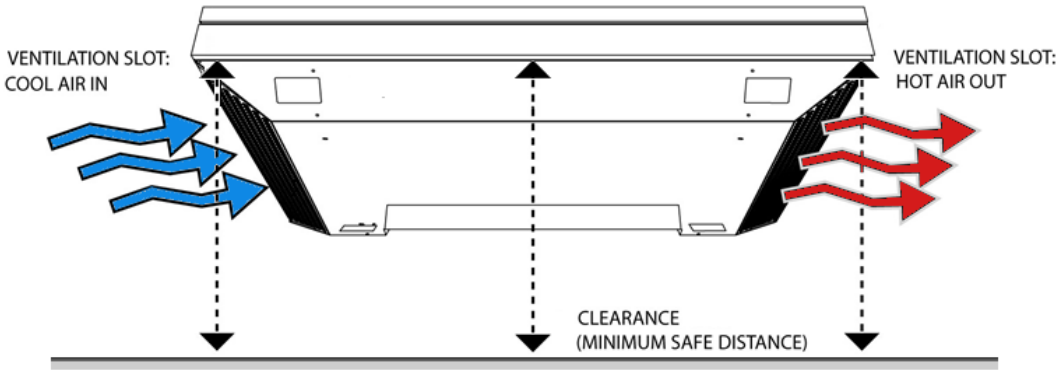
4.3 Ventilation and heat management

Heat build-up reduces lifetime and operating safety of the device and can cause hardware failure. It is essential that you control a Cell's exposure to heat.

Ventilation and heat management are especially important in a multiple-Cell installation. You must ensure there is adequate airflow (cool air in and hot air out) through the ventilation slots on each Cell. In particular, ensure that hot air vented from Cells is directed away from the cool air intakes of adjacent Cells; see the example in section 5.4.1.

To ensure proper heat management, review the following checklist.

Item	Correct status
Ventilation	Ensure the Cell is installed in a well-ventilated environment and in a well-ventilated enclosure (if enclosure is used). The device is not covered with clothes, papers, props or other materials which can prevent the proper ventilation of the device.
Operating temperature	Ensure that temperature is within the normal operating limits in section 4.2.
External factors	Ensure there are no external factors heating the Cell above the temperature limits. External factors can include for example direct sunlight, spotlights or other heat emitting devices.
Installation clearance	Ensure the gap (or clearance) between the Cell and wall, or between the Cell and base of the enclosure, matches the *minimum safe distance.* ▪ For single Cells, this is at least 10cm (4 in) ▪ For multiple-Cell displays, this is at least 15cm (6 in) *See also section 5.2.2 and section 5.4.*



Ventilation slots and clearance. The clearance must match the minimum safe distance.

If you have specific installation requirements (for example, if your installation design requires a reduced clearance), please contact MultiTaction Support for advice: www.multitaction.com/support-services

See also the warnings on page 18.

⚠ Caution: *Make sure that the cooling and air ventilation properties of the installation location meet the requirements of the Cell.*

Keep the ventilation slots free of blockages at all times. Leave enough space around the Cell to allow proper air circulation and cooling between the display and walls or panels:

⚠ Avertissement*: S'assurer que le refroidissement et la ventilation du local d'installation sont conformes aux exigences de la Cellule.*

Faire en sorte que les évents ne soient jamais obstrués. Laisser suffisamment d'espace autour de la Cell, pour permettre à l'air de circuler et au refroidissement de se faire entre l'écran et les murs ou panneaux:

5 Get the Cell into position

The MultiTaction Cell must be installed before use. Follow one of the four installation guidelines:

- VESA Mount Installation; see section 5.1.
- Custom Install; see section 5.2.
- Desk Stand Installation; see section 5.3.
- Multiple-Cell Installation; see section 5.4.

Before installing, ensure the installation and usage environment is suitable for the MultiTaction Cell; see section 4.

5.1 VESA mount installation

The MultiTaction Cell complies with the VESA Mounting Interface Standard (MIS-F) by having four M8 screw holes on its back panel. You can install the Cell to any pedestal or wall mount that conforms to the VESA Mounting Interface Standard.

Important: *When mounting a Cell, remember that:*

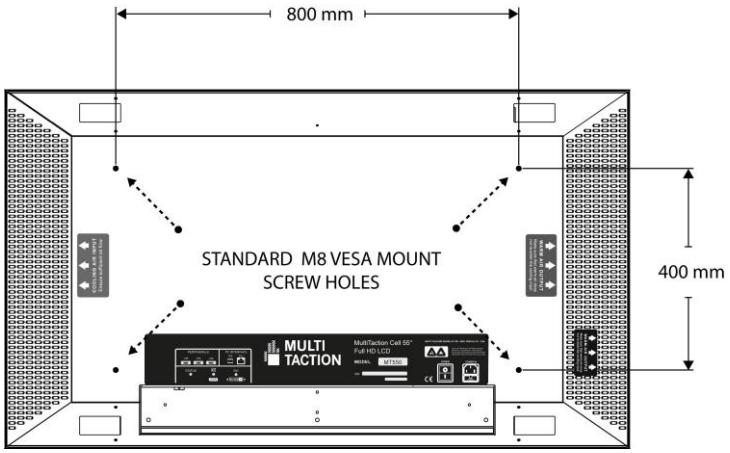
- *Screws can enter the panel only up to 8 mm in depth. (See also the thread length requirement in section A.9 VESA Mount)*
- *Cells are heavy, weighing 40 or 41kg depending on the model.*

To mount a Cell:

1. Locate the Standard M8 VESA Mount Screw holes on the back panel of the Cell.
2. Select the VESA Mount installation location and install the Cell according to the VESA Mounting Interface Standard.

   Note the cooling and ventilation requirements in section 4.3.

The VESA mount installation is now complete.



MT553 VESA mount screw holes

5.2 Custom installation

A Cell can be installed as a table in a horizontal or tilted position, or in a custom enclosure.

Cells have an alignment rail on the top, bottom and sides. Use the rail to align the Cell in a custom enclosure with M5 screws and alignment rail bits. A Cell is attached to the enclosure by the VESA screw holes.

A simplified custom setup would consist of sturdy supporting legs on each side, firmly wrapping the Cell and keeping it at the *minimum safe distance 15 cm (6 inches)* from the top of a stable surface.

To complete a custom installation, follow these steps:

1. Prepare the custom table or enclosure for the Cell.

   Ensure there is a gap of at least 15 cm or 6 inches (ie, the minimum safe distance) below the Cell to allow sufficient ventilation (cool air in, hot air out). See the diagrams in section 5.2.2 and section 5.2.3.

2. Locate the M8 type screw holes (VESA) and the Cell alignment rail.

3. Firmly attach the Cell to the custom table or enclosure using M8 screws and the Cell's VESA screw holes. Ensure the Cell is correctly aligned.

Danger

⚠ *Never install a Cell inside a fully enclosed space! Risk of device overheating, which can lead to critical hardware failure and personal injuries.*

Ensure the custom enclosure design complies with the minimum safe distance requirement ie, there must be a gap of 15 cm or 6 inches between the Cell underside and the base of the enclosure to allow sufficient air ventilation (cool air in, hot air out).

Danger

⚠ *Ne jamais installer la Cellule dans un espace entièrement los! Risque de surchauffe du dispositif pouvant causer une panne critique du matériel et des blessures corporelles.*

S'assurer que les dimensions et le montage du dispositif d'utilisation conforment avec les distances de sécurité requises, à savoir 15 cm (6 pouces) au dessus d'une surface stable, et que la Cellule a son propre système de ventilation (entrée d'air frais, sortie d'air chaud).

Caution

⚠ *Loosely installed Cell may fall and cause personal injuries.*

Attach the Cell properly by the VESA screw holes. If the Cell is installed in a public environment, check the condition of the enclosure and attachments regularly.

Avertissement

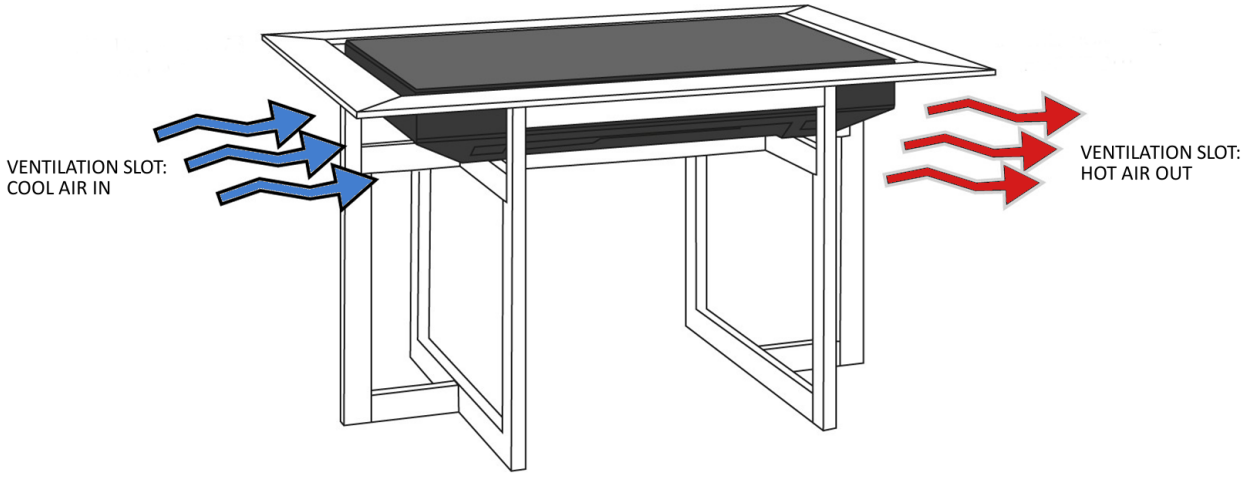
⚠ *Une Cellule qui n'est pas montée solidement peut tomber et causer des blessures corporelles.*

Attacher la Cellule correctement par les trous de vis VESA. Si la Cellule est installée dans un milieu public, contrôler régulièrement la condition du boîtier et des fixations.

Notes

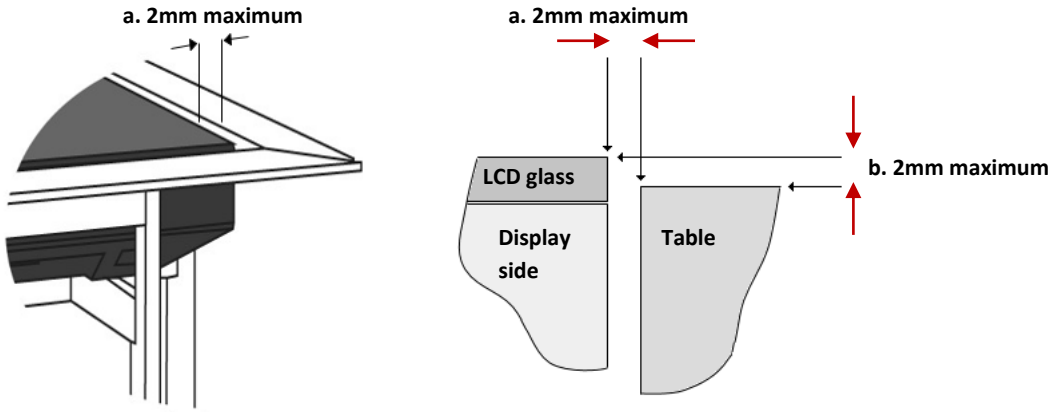
- *Any unauthorized modification of the Cell will void warranty.*
- *The front glass may be changed in coordination with MultiTouch. This requires an explicit warranty exception.*

5.2.1 Custom table with installed Cell



Custom table with installed Cell

Ensure that the clearances between the display side and table, and between the LCD surface and surrounding table surface, do not exceed 2mm. The table should protect the edge of the display.

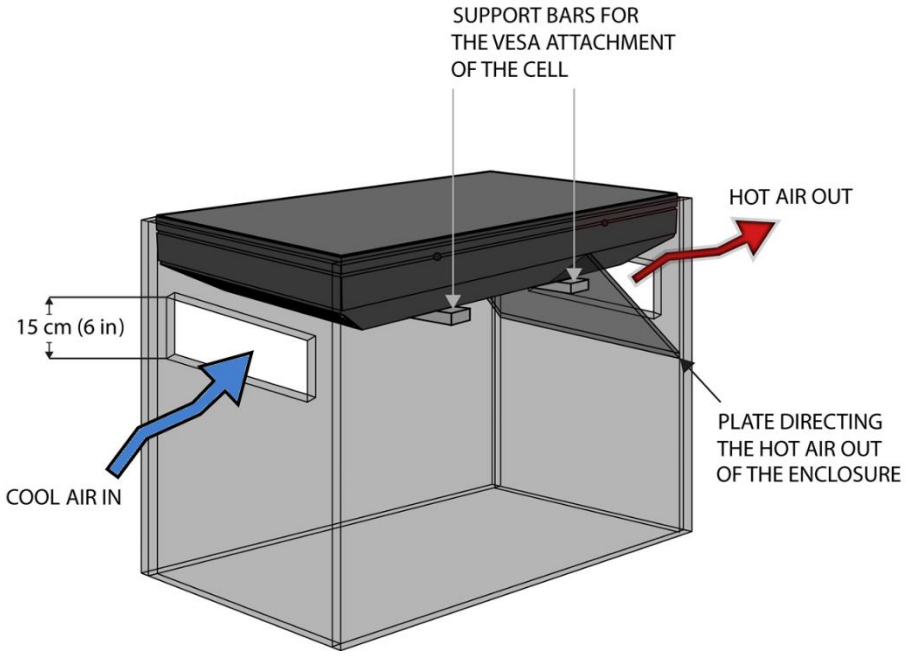


MT553UTB and MT555UTB: Maximum table clearances

a. *Maximum 2mm gap between display side and table*

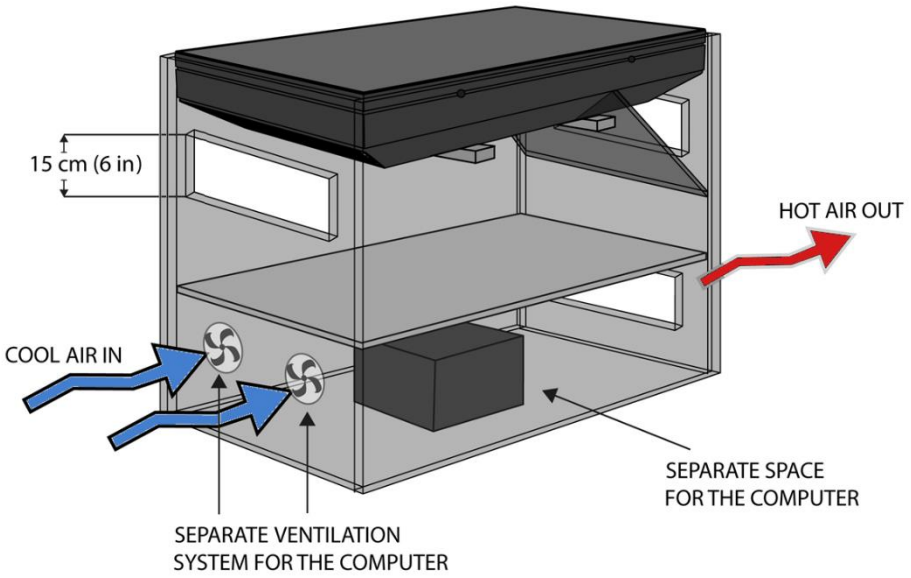
b. *Maximum 2mm difference between LCD surface and table surface*

5.2.2 Custom enclosure with installed Cell



Custom enclosure with installed Cell. Note the 15cm ventilation gap below the Cell to allow sufficient cool air to flow in.

5.2.3 Custom enclosure with separate space and ventilation system for computer unit



Custom enclosure with separate space and ventilation system for computer unit. Note the 15cm ventilation gap below the Cell to allow sufficient cool air to flow in.

5.3 Desk stand installation

The MultiTaction Cell comes with a pair of desk stands. Desk stands ensure safe and stable installation when the Cell is positioned horizontally or vertically on a sturdy desk or table.

1. Choose a sturdy desk for the installation. Ensure the desk can bear the weight load of the Cell (45 kg / 100 lbs) and that the desk will not tip over when the Cell is installed.

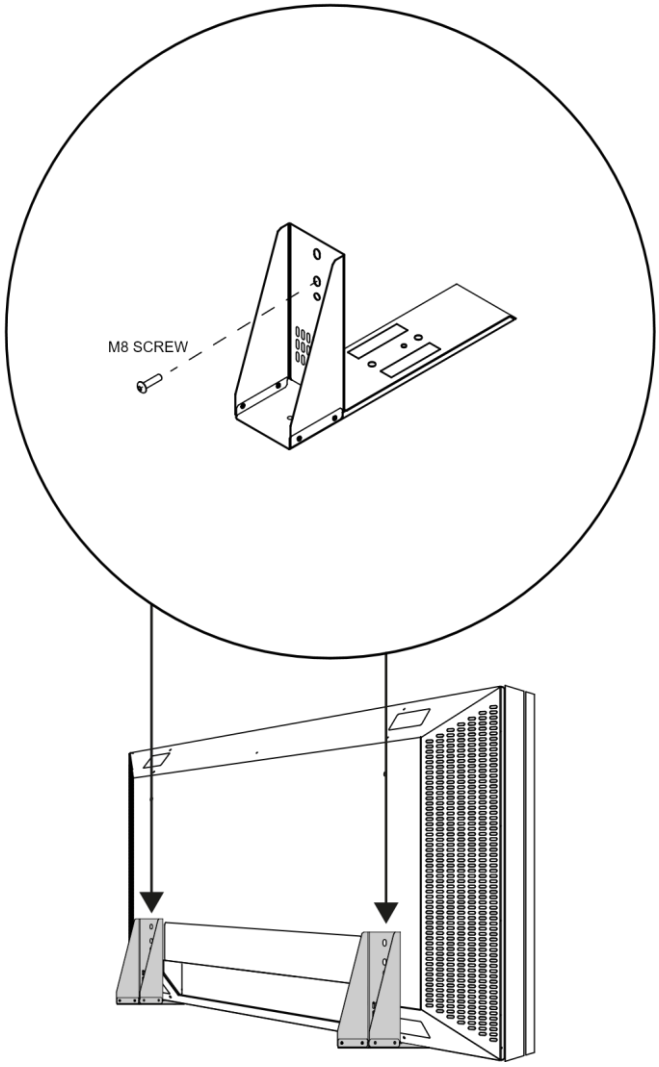
   Caution: *Using a desk that is too weak, installing the desk stands incorrectly, or installing into an incorrect position, causes a risk of the Cell tipping over, property damage and personal injuries. To avoid this risk, choose a desk with a heavy enough structure for the installation, install the desk stands according to this manual and ensure that desk will not tip after the installation.*

   Avertissement: *Si la table utilisée est d'une structure trop légère ou si celle-ci est mal soutenue ou installée dans une mauvaise position, la Cellule risque de se renverser et de causer des dommages matériels ou personnels. Pour éviter ce risque, choisir une table de structure suffisamment solide pour l'installation, monter la table selon les instructions de ce manuel et s'assurer que la table ne puisse pas s'incliner après l'installation.*

2. Depending on the Cell orientation (horizontal or vertical), place the cell on the stands on the desk and attach them with M8 screws to the rear of the display.

   Caution: *To minimize the risk of tipping, do not place the stands on the edge of the desk.*

3. Lift the Cell on the desk stands (two people required) and attach it to the stands with M8 screws (VESA screw holes). Align the Cell.
4. Ensure that the desk will not tip when the Cell is installed on the desk stands.

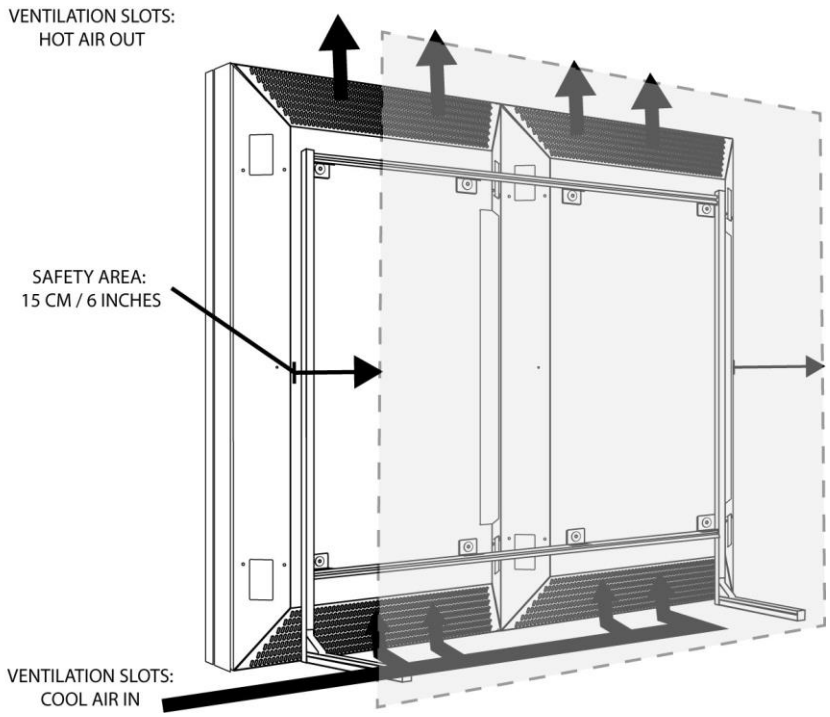


Desk stands and M8 screws

5.4 Multiple-Cell installation

Using the VESA mounts and a custom sturdy metal frame, firmly and securely install two or more Cells together to operate in a multiple-Cell configuration.

Ventilation and heat management are especially important in a multiple-Cell installation. You must ensure there is adequate airflow (cool air in and hot air out) through the ventilation slots on each Cell. Leave enough space between the Cells and the wall to allow proper air circulation and cooling. Also, ensure that hot air vented from Cells is directed away from the cool air intakes of adjacent Cells.



Horizontal Installation of multiple Cells

Caution

If Cells are installed too close to each other or too close to walls or panels behind the Cells, there is a risk of device overheating. This can cause critical hardware failure and personal injuries. Leave enough space between the Cells and any walls or panels to allow proper air circulation and cooling. The minimum safe distance for a multiple-Cell installation is 15 cm (6 inches).

Avertissement

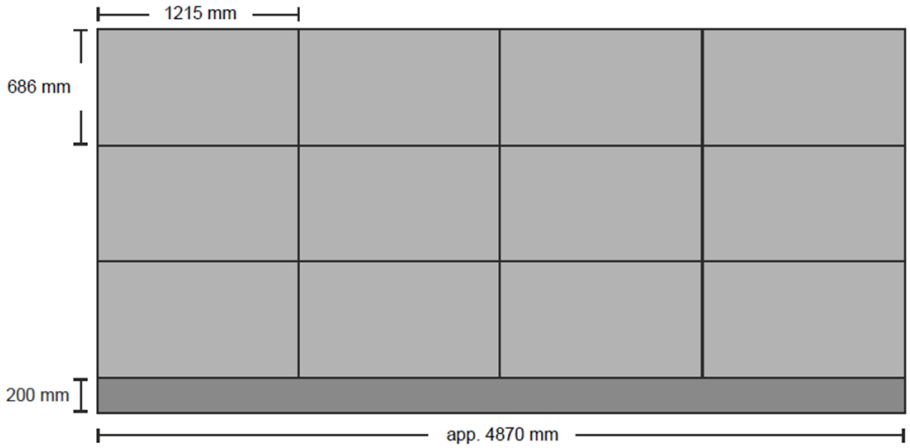
Les Cellules qui sont installées trop proches les unes des autres ou trop près des murs ou des panneaux arrières, risquent de surchauffer le dispositif, ce qui peut causer une panne critique du matériel et des blessures corporelles. Laisser suffisamment d'espace entre les Cellles et les murs ou les panneaux derrière les Cellules pour permettre une bonne circulation d'air et un bon refroidissement. La distance de sécurité requise pour une installation multiple-Cell est de 15 cm.

5.4.1 Display wall installations

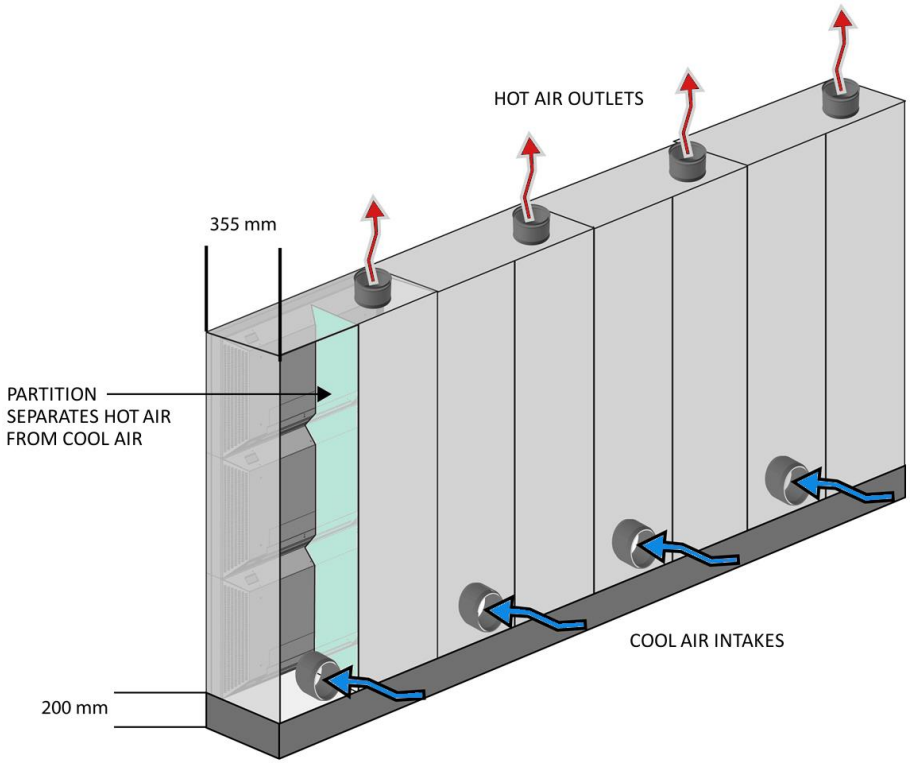
When planning and implementing display wall installations, remember the technical requirements for heat management of the Cells (section 4.3).

Ensure that Cells located in one ventilation space have a sufficient intake of cold air. (The typical use for MT553UTB and MT555UTB models is 220 m^3 per hour per Cell.)

The following example of a 4x3 display wall of MT553UTB Cells provides general guidelines for the design procedure. Each column of three Cells has its own cool air intake and hot air outlet with a physical partition separating the intake and outlet.



Front view: 4x3 display wall of MT553UTB Cells



Isometric view: 4x3 display wall of MT553UTB Cells
Physical partitions separate the cool air intakes from the hot air outlets

6 Connect the Cell

The Cell connection panel is located at the back of the device; see section 2.2.

Warning: *People can trip over loose and hanging cables, causing personal injuries and damage to the Cell. Route and fasten all cables out of the way.*

Avertissement: *Les gens peuvent se trébucher sur des câbles non fixés ou pendants, ce qui peut causer des dommages personnels ou endommager la Cellule. Acheminer et fixer tous les câbles à l'écart*

6.1 Connections to the mains power supply

To protect Cells from voltage spikes, we recommend that you:

- Always connect Cells to the mains power supply using a 3-wire cord and power strip with a protective earth. *Failure to do so may cause an electric shock when using the Cell!*
- When connecting other equipment to the Cell, make sure all equipment is disconnected from the mains. Failure to do so may damage the Cell through ground leakage currents from and to other devices.
- Do *not* connect more than two Cells to a single power strip when connecting them to the mains power supply.
- Power up the Cells one at a time in multiple-Cell installations.

  Note: *Here, a power strip refers to a block of sockets that allows multiple devices to be powered from a single electrical socket. Other common names for a power strip include power bar, power adapter and socket extension, plus many other variations.*
- Restrict the number of Cells per mains fuse. The number depends on the rating of the mains electricity supply:
  - 230VAC: Do not connect more than *four* Cells per mains fuse.
    Countries with 230VAC power include most of Europe, Asia, Africa, and South America, plus Australia and New Zealand.
  - 100-120VAC: Do not connect more than *two* Cells per mains fuse.
    Countries with 100-120VAC power principally include the US, Canada and Japan.

For Cell power consumption details, see section A.5 Power consumption.

6.2 Connections to an external application computer

MultiTaction Cells require an external application computer with two cable connections. While the Cell and application computer are disconnected from the power supply, connect:

- An Ethernet cable from the Cell to the application computer

  This Ethernet cable transmits finger locations and marker information to the application computer. The cable is generally connected to a switch on the LAN so the Cell can use the connection to download firmware updates as necessary. Alternatively, you can connect the Cell directly to the application computer. We do not recommend a wireless connection because of the latency it introduces.

- **A DVI cable from the Cell to video outputs on the application computer**.

  This cable enables the video image from the computer to be displayed on the Cell.

Ensure there is no stress or tension on the connected cables. After connecting all cables, connect the Cell and computer to the mains supply.

Note: *For a description of the Cell connection panel, see section 2.2.*

6.3 Connections to the internet

We recommend that all Cells have an internet connection. For this reason, Cells have an Ethernet port for connecting to your default gateway or an external computer; see section 2.2.

If you intend to display Web applications on your Cells, your *application computer* must have an internet connection.

Note: *See also section 8.3 for details about configuring network settings for the Cell's tracking engine.*

6.4 USB mouse and keyboard

To show the on-screen display (the OSD; see section 8.2), you will need a mouse and keyboard attached to USB ports on the Cell.

For convenience, you may prefer to attach a wireless mouse and keyboard so you can stand in front of the Cell while using the on-screen display.

However, after the OSD appears on-screen, you can configure it to accept finger input. That is, you can use your fingers to configure settings in the OSD; see section 8.2.2.

Note: *You do not need a mouse or keyboard to use the OSD when the Cell first starts up. During startup, the OSD displays automatically and you can configure it to accept finger input.*

7 Start up the Cell

Follow these steps when you start up the Cell for the first time:

1. Connect the power cable and turn the Cell on with the power switch on the back of the device (see section 2.2).
2. Verify the boot sequence.

   When you turn the Cell on, confirm that events display in the following sequence:

   a. The MultiTaction 'M' logo briefly appears:

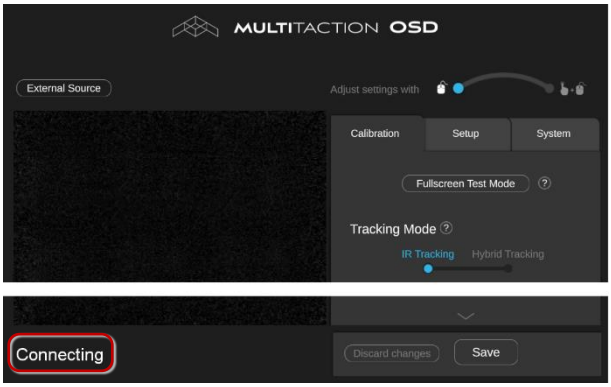
   

   b. Boot text briefly appears.

   c. The MultiTaction 'M' logo briefly re-appears, this time with a progress bar:

   d. The on-screen display (OSD) briefly appears with a 'Connecting' message:

   e. The boot sequence completes when the full OSD appears.

3. Now use the OSD to configure the tracking engine; see section 8.

8 Set up the tracking engine

The *tracking engine* runs automatically on the Cell's internal computer. It processes image data captured by the Cell's camera matrix as users touch the screen. The tracking engine then outputs this information as *tracking data* (also called *touch data*) to touch-enabled applications. Tracking data can include fingertip locations, infrared pen locations, plus marker locations and marker code values.

For stackable Cells, these touch-enabled applications run on an external *application computer*. The tracking engine must therefore be able to exchange tracking data with the application computer over an IP network.

Internally, the tracking engine comprises a set of *trackers* that exclusively process different types of tracking data. Currently, there are separate trackers for fingers, markers and infrared pens. You can fine-tune these trackers, and you can enable or disable individual trackers.

To set up the tracking engine, you need to calibrate the cameras and configure the individual trackers. For stackable Cells, you also need to configure network settings for the tracking engine to enable it to communicate with the application computer. For all these setup tasks, you use the on-screen display (OSD).

8.1 Which firmware version?

Instructions in the following sections refer to Cell firmware version:

2.0.8-taction 17 or later

8.2 Using the OSD

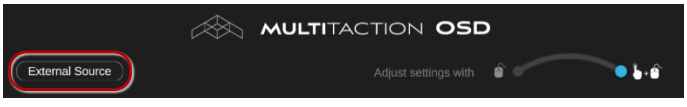
MultiTaction Cells include an on-screen display (OSD) running on the internal computer. You primarily use the OSD to configure the tracking engine so that tracking data can be received by, or transmitted to, the application computer. You can also use the OSD to view Cell details such as its MAC address and current firmware version.

8.2.1 Display the OSD

A Cell has two sources for the image that is displayed on the screen: the on-screen display (OSD) or the display on an application computer (the 'external display').

MultiTaction Cells now have an integrated DVI switch that allows the LCD image to come from the internal computer or an external computer.

- To show the OSD and hide the external display, you need a mouse or keyboard attached to a rear USB port on the Cell; see section 6.4. Simply press the keyboard or move the mouse to show OSD. (Closing the external display is similar to turning off a screensaver in Windows.)
- To hide the OSD and return to the external display, click the External Source button in the top left corner of the OSD.

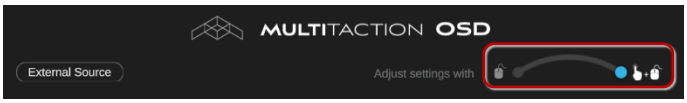


OSD External Source button

8.2.2 Specify mouse or finger input

You can use a mouse or your fingers to configure settings in the OSD. By default, the OSD only accepts mouse input. To configure the OSD to also accept finger (touch) input:

1. Display the OSD.
2. Drag the 'mouse or finger' slider from  Mouse Only to  Finger & Mouse.



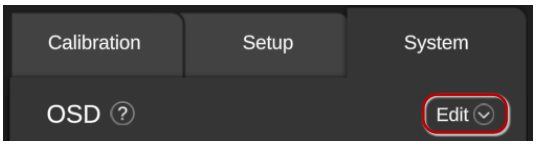
OSD 'mouse or finger' slider

Note: *You must re-specify Finger & Mouse input each time you launch the OSD.*

8.2.3 Specify an OSD timeout

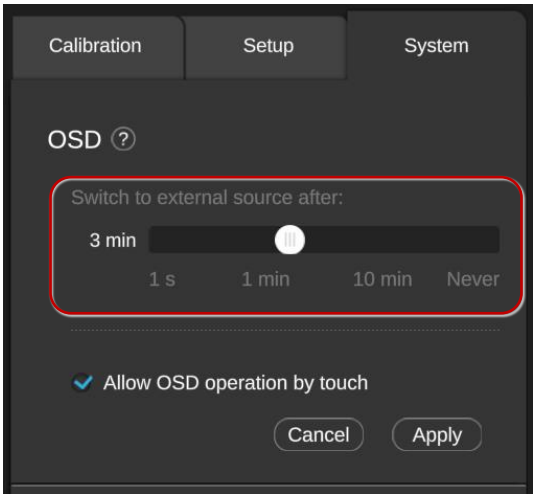
You can also specify a timeout to hide the OSD automatically:

1. Go to the OSD section in the OSD System tab.
2. Tap the Edit dropdown.



OSD System tab, OSD section, Edit dropdown

3. Set the slider to the required timeout. Then tap Apply.



OSD System tab, OSD section, timeout slider

4. Tap the Save button at the bottom of the System tab.

8.2.4 Save or undo changes to OSD settings

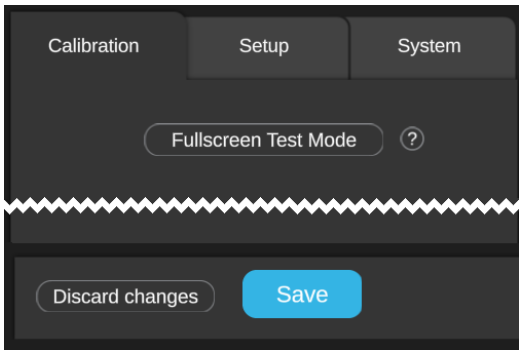
OSD settings can be adjusted in real time, but the new values are not saved automatically. Instead, you must manually save changes to parameter values. If you do not save the changes, the Cells reverts to the previous values when it next restarts.

To save changes to OSD settings:

1. Tap the Save button. You may need to scroll to the bottom of the current tab to see this button.
2. The Cell saves the new settings to its internal memory. This can take up to 20 seconds.

   Warning: *Do not turn off the Cell while saving the new settings!*

To undo changes to OSD settings, tap the Discard Changes button.



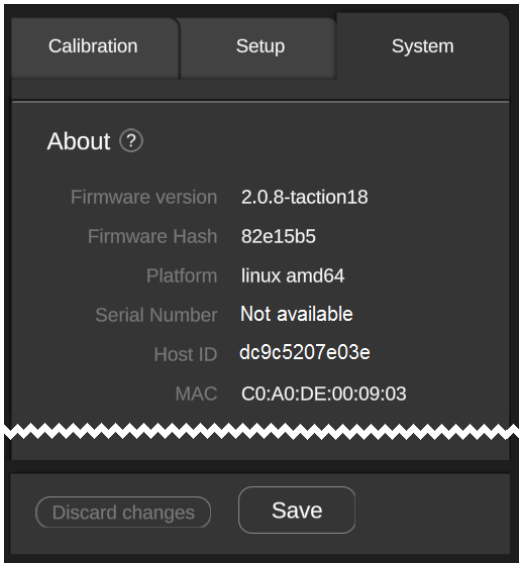
Save button and Discard Changes button

8.2.5 View 'About' information

You can use the OSD to view system information for the Cell (for example, firmware version, platform details, and MAC address).

To view system information:

1. Show the OSD.
2. Scroll down to the About pane of the System tab.



OSD System tab, About pane

8.3 Configure the network settings

The tracking engine needs an IP address to allow communication between itself and the application computer. You can manually assign a static IP address or you can specify that your DHCP server assigns an IP address automatically.

Note: *This task primarily applies to stackable Cells, although it is also possible for an external application computer to receive tracking data from an embedded model.*

8.3.1 Assign a Cell IP address

To set up an IP address:

1. Display the OSD and tap the Setup tab.
2. Go to the Network pane and set the Configuration slider to DHCP or Manual (see the screenshot below).

   If you select Manual, specify the following:

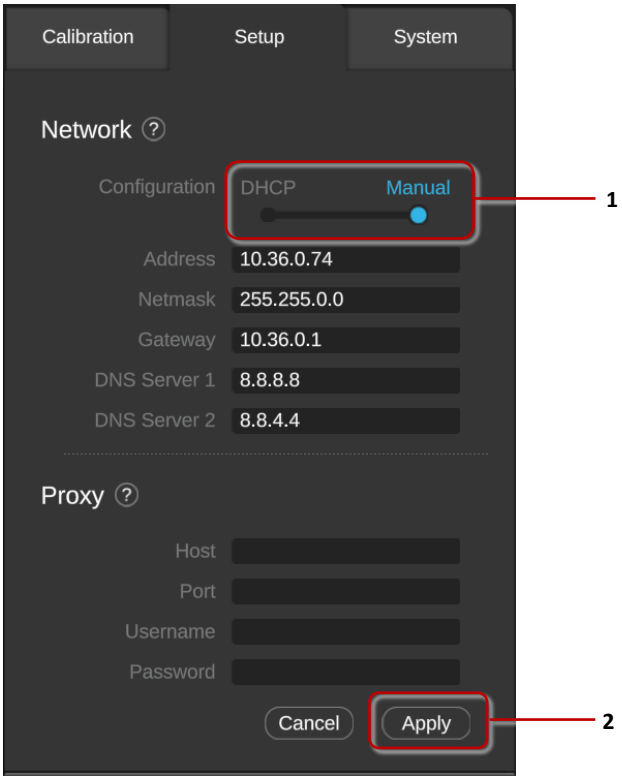
   Address: Enter the IP address.

   Netmask: Enter the subnet mask for your network.

   Gateway: Enter the IP address of the default gateway.

   Depending on your network requirements, you may also need to specify preferred and secondary DNS servers, and proxy server details.

   Warning: *If you manually change a Cell's IP address after you have set up the tracking engine, you must update config.txt with the new IP address; see section 9.5.*
3. Tap the Apply button.



*OSD Setup tab, Network pane. **1** DHCP or Manual slider. **2** Apply button.*

4. Remember the tracking engine's IP address. You will need to supply this address when configuring the application computer; see section 9.5.
5. Tap the Save button to save the new network settings.

   Tip: *You may need to scroll to the bottom of the Setup tab to see the Save button.*

The tracking engine now starts using the specified IP address.

8.3.2 DHCP or manual IP addresses?

By default, MultiTaction Cells use DHCP for network configuration.

If you choose to continue using DHCP, you can set up your DHCP server to always assign the same network address to a Cell by filtering on the Cell's MAC address. This enables allows the Cell to operate in DHCP mode while making it available to the application computer through a fixed network address. For details about finding a Cell's MAC address, see section 8.2.5.

In some cases, you may need to manually assign an IP address to a Cell. For example, you may want to connect Cells to an application computer directly, without using a DHCP server. If you assign a manual IP address, the address cannot be on the prohibited subnet; see section 8.3.3.

8.3.3 Prohibited subnets

MultiTaction Cells have an internal computer that communicates with the internal tracking engine over Ethernet. Consequently, the following subnets are prohibited ie, reserved for Cell internal use only. Do not use an address on the prohibited subnet when manually assigning a network IP address to a Cell.

The prohibited subnet is 192.0.2.x.

In stackable Cells, the OSD prevents you from manually assigning an IP address in the forbidden range. However, your DHCP server may do so. If your DHCP server does assign an IP in the forbidden range to a Cell, the Cell will be unresponsive when users touch the screen and you will need to manually assign a legitimate IP address.

To see which IP address has been applied to your Cell and, if necessary, manually assign a legitimate IP address, check the OSD Network settings; see section 8.3.1.

8.4 Adjust the tracking

8.4.1 Geometry calibration

MultiTaction Cells utilize a camera matrix for tracking objects on the Cell display surface. During transport and installation, the physical position of the cameras may move a few millimetres if, for example, there was rough handling during transit. This will require a recalibration of the geometrical alignments of the cameras. Note that finger tracking usually works even without geometry calibration, but for marker tracking it is often necessary.

Preparations

You need a white calibration sheet that is big enough to cover the whole display. One calibration sheet is included with each MultiTaction Cell.

Tip: *Ensure the calibration sheet is firmly in place to minimize light leakage onto the screen.*

Calibration procedure

To perform the calibration:

1. Connect a keyboard to the Cell.
2. Switch to the on-screen display (OSD).
3. In the OSD, go to the Calibration tab.
4. Go to the Geometry Calibration pane and tap the Calibrate button to start the calibration procedure.
5. Place the white calibration sheet on top of the display. You have 15 seconds to do this before the actual calibration process begins.

   Tip: *Minimize light leakage onto the screen by taping the calibration sheet in place on the Cell screen. Use a good quality paper tape that adheres well but leaves minimal adhesive residue when removed.*
6. Wait up to five minutes for the calibration to complete.
7. When the calibration is complete, tap the Save button.

   Tip: *You may need to scroll to the bottom of the Calibration tab to see the Save button.*

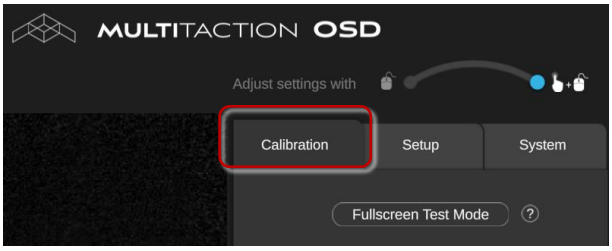
Important! *Retain at least one of your Cell calibration sheets! Do not throw them all away after setting up your video wall. There may be occasions in the future when you need to recalibrate individual Cells.*

8.4.2 Adjust the tracking parameters

After the camera geometry calibration is complete (see section 8.4.1), you can test the touch tracking and, if necessary, adjust the touch tracking parameters. Your goal is to make touch tracking as accurate as possible.

There are separate tracking parameters for different tracking features: fingers, markers and pens. The configuration procedure is the same in each case:

1. Display the OSD and tap the Calibration tab.

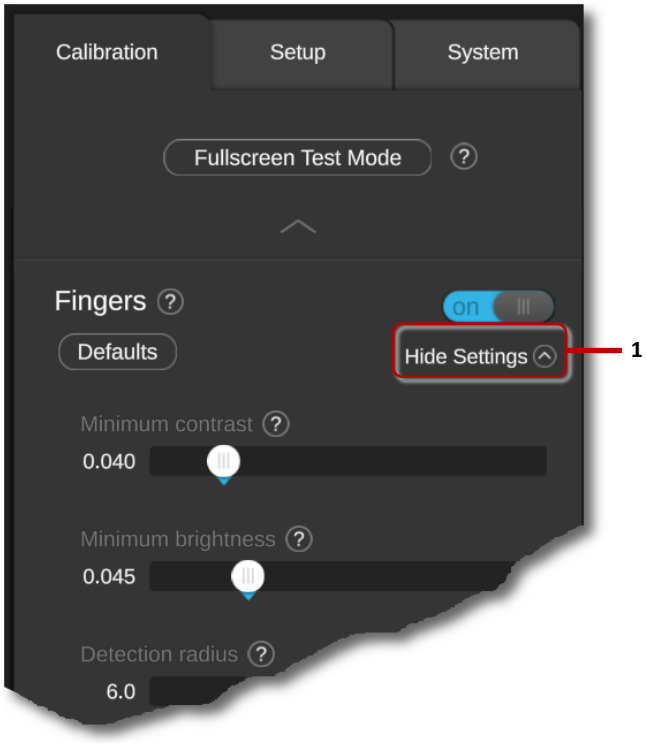


OSD Calibration tab

2. Go to the pane for the tracking feature you want: Fingers, Markers or Pens.
3. Turn on the tracker by moving the On-Off slider to On. For example:

OSD Calibration tab, Markers pane, On-Off slider

4. By default, the tracking parameters are hidden. Tap the Edit Settings button to display the available parameters
5. Adjust the parameters as required. For parameter details, see:
   - Finger tracking: section 8.4.3
   - Square marker tracking: section 8.4.4
   - Infrared pen tracking: section 8.4.5
6. (Optional) To declutter the OSD, tap the Hide Settings button to hide the tracking parameters for the current tracking feature.



OSD Calibration tab; Fingers section, example tracking parameters.
***1** Hide Settings button.*

7. Tap the Save button to save the new tracking parameters.

   If you do not save, any changes to the tracking parameters will be lost when the tracking computer next restarts.

   Tip: *Scroll to the bottom of the Calibration tab to see the Save button.*

8.4.3 Finger tracking parameters

The most important tracking parameters for fingers are listed below.

- **Minimum contrast** defines how bright a fingertip should be, compared to its immediate surroundings.

  Lower this value to make tracking more sensitive; increase this value to make tracking less sensitive. The typical value is 0.04. You rarely need to change this value.

- **Minimum brightness** is the most important parameter for finger detection sensitivity. It defines how bright finger tips are expected to be. Due to subtle differences between the Cells, you may need to adjust this parameter from its default value.

  Lower this value to make tracking more sensitive; increase this value to make tracking less sensitive. The typical value is between 0.04 and 0.09.

- **Detection radius** defines the expected radius of a fingertip, measured in camera pixels. This parameter is almost always set to 8. The useful range is between 6 and 9. Smaller values imply a user has smaller fingers; larger values imply larger fingers.

  Increasing this value makes tracking more sensitive, but you rarely need to adjust this parameter.

The following tracking parameters for fingers are also available. You rarely need to adjust these parameters.

- **Sensitivity** determines how readily possible fingertips are classified as fingers. The useful range for this parameter is between 1.0 and 1.5.

  Increasing this value makes tracking more sensitive, but you rarely need to adjust this parameter.

- **Keep-alive** defines how long lost fingers are maintained in the finger list (measured in frames). The useful range for this parameter is from 2 to 4 frames.

  Increasing this value will reduce the tracking gaps when a finger can be lost, but will also increase latency when fingers are lifted from the screen.

- **Position filter strength** filters the raw fingertip coordinates to reduce unwanted jitter in reported tracking coordinates. The filter only applies to slow hand movements. Increasing the filter strength will make tracking less jittery, but will also increase latency ie, tracking is slightly slower.

  The typical range for this parameter is between 0.9 and 1.0, but in some cases you may need to set it to zero to eliminate filtering completely.

- **Position filter threshold** defines the effective radius of the position filter. Lower values result in less filtering and faster tracking; higher values result in more filtering so tracking is smoother, but with higher latency ie, tracking is slightly slower.

  The typical range is between 20 and 50.

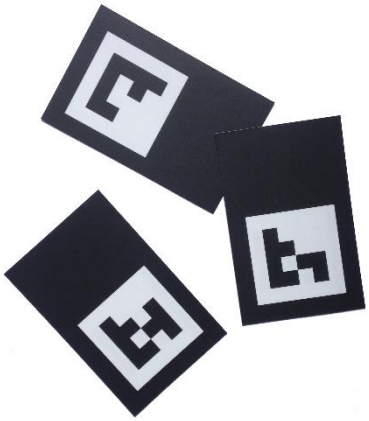
- **Lens compensation** is an internal parameter for compensating certain lens effects. You rarely need to change this value.

- **Background keep** determines how fast the *background model* adapts to screen changes caused by touch events. Lower values mean faster adaptation. A value of 1 means no adaptation ie, the background model is never updated.

  Note: *The 'background model' refers to the Cell cameras' default view of the touch screen when there is no touch activity. The model also takes account of ambient light conditions.*

8.4.4 Marker tracking parameters

Markers are square 2D barcodes, each with a unique marker code, printed onto paper or card and used on the Cell screen. Applications can access the marker information through the Cornerstone API, and also through TUIO and XML streams of tracking data.



Example markers printed onto the back of a business card

There are minor differences between the infrared sensitivity of the Cells, so you may need to adjust the sensitivity of the marker tracker. The most important tracking parameters for markers are listed below.

- Size specifies the size of marker that can be recognized by the tracker. In turn, the marker size determines the maximum number of unique marker codes that the tracking engine can support. Three marker sizes are supported:

Marker size	Example	Number of unique codes	Minimum length and width of physical marker
3x3		32	3.5 x 3.5 cm
4x4		4,096	4.0 x 4.0 cm
5x5		2,097,152	4.5 x 4.5 cm

Note 1: *'Marker size' refers to the number of rows and columns in the 2D barcode. It does not refer to the length and width of the physical marker. The marker size should match the division value used to generate the marker with the MarkerFactory application.*

Note 2: *The minimum length and width of a physical marker include the marker's outer black and white borders.*

- **Minimum brightness** is the most important value for marker tracking. It defines the expected brightness of white areas on a marker ie, the threshold for detecting the white areas. Typical values are between 40 and 60.
- **Keep-alive** defines how long lost markers are held in memory before they are finally removed from the marker list. The time period is measured in frames. The useful range for this parameter is between 5 and 20 frames.

  Increasing this value makes tracking more stable but will also increase latency when markers are lifted from the screen. Lower values may cause marker tracking to flicker.

The following tracking parameters for markers are also available. You rarely need to adjust these parameters.

- **Maximum contour length** sets the maximum length of the marker edge, measured in pixels. In effect, this parameter limits the physical size of the markers. Higher values indicate larger circles; smaller values indicate smaller circles. For typical applications, 200 is a suitable setting.
- **Minimum contour length (relative)** controls the minimum size of the marker. Lower values mean that the markers can be very different sizes; higher values mean markers must have a more constant size.

  The typical range for this parameter is 0.5 to 0.8, although very low values rarely cause problems for the tracking. Values too close to 1.0 may cause poor tracking.
- **Minimum contrast** defines how bright a marker should be, compared to its immediate surroundings. Lower this value to make tracking more sensitive; increase this value to make tracking less sensitive.
- **Report delay** adds an artificial delay before reporting any newly detected marker.

  When a new marker is placed on the Cell screen, the tracker requires a few frames to stabilize the marker code value. Increasing the report delay helps the tracker to work reliably and makes the initial calculation of the marker code value more accurate, but it also adds latency ie, detection of new markers is slightly slower.
- **Maximum side ratio** controls how readily a slightly non-square marker is accepted as a square marker. This parameter measures the ratio of the longest marker edge to the shortest edge. This ratio is usually between 1.2 and 1.4.
- **Corner offset** specifies the thickness of the white frame around the marker's 2D bar code. Typical values are between 0.2 and 0.3.
- **Position filter strength** filters the raw marker coordinates to reduce unwanted jitter in reported tracking coordinates. The filtering only applies to slow marker movements. Increasing the filter strength will make tracking smoother (less jittery) but with higher latency ie, tracking is slightly slower. The typical range for this parameter is between 0.9 and 1.0.
- **Position filter threshold** defines the effective radius of the position filter. Lower values result in less filtering and faster tracking; higher values result in more filtering so tracking is smoother, but with higher latency ie, tracking is slightly slower. Typical values are between 20 and 50.

- **Allow code change** determines how the tracking engine handles a re-evaluation of a marker's unique code.

  A marker's unique code can be re-evaluated if the tracking engine's first attempt to identify the code is incorrect (for example, because the printed barcode is indistinct or worn). Later in the session, after the marker has been presented to the screen several times, the tracking engine correctly identifies the marker's unique code. The tracking engine must now determine how to handle this re-evaluation (the 'code change').

  The correct handling depends entirely on the application running on the Cell(s):

  - If you set the parameter to 1, the tracking engine allows the code change while keeping the marker's touch history (its touch IDs) unchanged for the current session. For some applications, this permits the application to continue running.
  - If you set the parameter to zero, the tracking engine disallows the code change. It deletes the marker's touch history (its touch IDs) for the current session. In effect, it interprets the code change as a wholly new marker (with a new touch ID). For some applications, this will terminate the application.

8.4.5 Infrared pen tracking parameters

The most important tracking parameters for pens are:

- **Minimum brightness** sets the threshold for detecting the tip of the pen. Typical values are between 220 and 255.
- **Maximum contour length** sets a limit on the length of curve that contains the pen LED spot. Increase this setting from the default if the pen has a very wide infra-red beam. For typical applications, the default length of 150 pixels is suitable.

The following tracking parameters for pens are also available. You rarely need to adjust these parameters.

- **Minimum contour length (relative)** sets a limit to the length of the curve that contains the pen LED spot. In effect, this parameter limits the physical size of the pen. Higher values indicate larger pens; smaller values indicate smaller pens.
- **Keep-alive** defines how long lost pens are maintained in the pen list (measured in frames). The useful range for this parameter is from 2 to 4 frames.

  Increasing this value will reduce the tracking gaps when a pen can be lost, but will also increase latency when pens are lifted from the screen.
- **Strictness** defines how circular the pen shape must be. Higher values mean the tracker will only recognize circular pen tips. Lower values allow the tracker can detect ellipses and other shapes.
- **Horizontal tip offset** applies an offset to the pen tip location.
- **Vertical tip offset** applies an offset to the pen tip location.
- **Position filter strength** filters the raw pen coordinates to reduce unwanted jitter in reported tracking coordinates. The filter only applies to slow pen movements. Increasing the filter strength will make tracking less jittery, but will also increase latency ie, tracking is slightly slower.

- **Position filter threshold** defines the effective radius of the position filter. Lower values result in less filtering and faster tracking; higher values result in more filtering, so tracking is smoother but with higher latency ie, tracking is slightly slower.
- **Mask threshold** is used to mask out any fingers holding the pen. It specifies the brightness threshold around the pen tip to prevent any other tracking from occurring.

8.4.6 Hybrid tracking

MultiTaction Cells include a feature called hybrid tracking, which is used to minimize the artefacts that are caused by ambient infrared lighting.

If you experience tracking problems which you suspect are caused by ambient infrared lighting, try using hybrid tracking. (For example, spurious touch events happen spontaneously in strong sunlight.)

When hybrid tracking is enabled, the Cell cameras operate at higher frame rate, alternating frame captures with and without a source of non-visible light (emitted by the infrared LEDs; see section 10.2.1). By comparing each frame pair, the Cell graphics system can eliminate spurious touch events and react only to real touches.

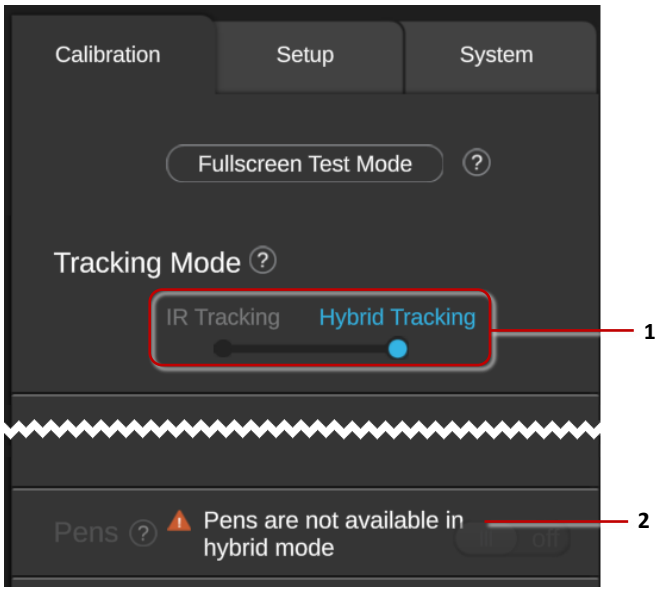
Note: *You cannot use infrared pens when hybrid tracking is enabled. That is, pen tracking is disabled.*

To enable hybrid tracking:

1. Display the OSD and tap the Calibration tab.
2. Go to the Tracking Mode pane.
3. Set the slider to Hybrid Tracking.

   A warning appears in the Pens section to remind you that infrared pens cannot be used while hybrid tracking is enabled.

   (To disable hybrid tracking, set the slider back to IR Tracking.)



OSD Calibration tab, Tracking Mode pane. 2 *Tracking mode slider.* 1 *Infrared pen warning.*

8.5 Create your own markers

You can generate your own 2D markers in .png format using the MarketFactory application. Alternatively, you can download ready-made sets of 3x3 markers and 4x4 markers in .pdf format Both methods are described in the following sections.

In either case, you can then print the markers onto paper or card and distribute to your users. Use any software that can open and print .png or .pdf files. But ensure you choose an appropriate size for your printed markers (section 8.5.3) and that you print them on suitable material (section 8.5.4).

You may also want to check section 14.1.3 for FAQs about finger tracking with markers and marker image burn-in.

8.5.1 Download ready-made square markers

Available for 3x3 and 4x4 square markers only.

Instead of creating your own square markers, you can download full sets of unique codes for 3x3 markers and 4x4 markers. The markers are in PDF format. Note that the physical size of the 3x3 markers is smaller than the 4x4 markers.

- **3x3 markers:** These markers support up to 32 unique codes.
  PDF versions of these markers are available to download here.
- **4x4 markers:** These markers support up to 4,096 unique codes.
  PDF versions of these markers are available to download here.

8.5.2 Create square markers

The MarkerFactory application is included with the Cornerstone runtime and SDK packages; see section 9.4 for installation instructions.

MarkerFactory is a command line application. Commands use this syntax:

```
MarkerFactory --code <n> [--division <n>] [--blocksize <n>] -o <file>
```

Where:

`--code <n>`

Specifies a unique marker, where `<n>` is the marker's code number. For example, this is the 4x4 marker for code 0130:

For 3x3 markers, code numbers run from 00 to 31.

For 4x4 markers, code numbers run from 0000 to 4095.

For 5x5 markers, code numbers run from 0000000 to 2097151.

`--division <n>`

Specifies the size of each marker, where `<n>` is the number of rows and columns. If this parameter is omitted, the marker size defaults to 5 (ie, a 5x5 marker).

`--blocksize <n>`

Specifies the size of each bar code block in pixels, where `<n>` is a number of pixels. The minimum block size is 16. If this parameter is omitted, the block size defaults to 32 pixels.

`-o <file>`

Specifies that the marker image is output to a file, where `<file>` is the file name. If `<file>` is omitted, the file name defaults to `code-XXXX.png` , where `XXXX` is the code number.

For example, to save the 4x4 marker above (code 130) to MyMarker.png, run this command:

```
MarkerFactory --code 0130 --division 4 -o MyMarker.png
```

8.5.3 Choose the marker size

When printing your markers, choose an appropriate physical size. Do not make them too small. The minimum physical size of the markers is dictated by how well the computer vision can determine the marker shape.

For each size of marker, the table below lists the minimum physical size and the number of unique markers they can represent:

Marker size (rows & columns)	Unique codes	Minimum physical size
3x3	32	3.5 x 3.5 cm
4x4	4,096	4.0 x 4.0 cm
5x5	2,097,152	4.5 x 4.5 cm
6x6	4,294,967,296	5.0 x 5.0 cm
On MultiTaction Cells, the width of a single bar code 'block' is around 5 mm.		

The physical sizes shown in the table correspond to the default values for marker tracking parameters in the OSD. If you generate markers with a different physical size, you will need to adjust the tracking parameters to compensate.

8.5.4 Print your markers on suitable material

When printing your markers, use suitable material to ensure the markers will work properly.

Usually, you would test your markers by simply printing them onto standard paper. But to test other materials, you must print a sample marker and place it on an actual Cell screen. Then observe the camera image visualization in the OSD and verify that the black regions of the marker appear dark on the camera image and the white regions appear white.

You *must* use a real Cell screen to verify the marker behavior because the Cell cameras work in the infrared spectrum, around 850nm wavelength. Consequently, it is possible to have visually black material on your marker that is completely reflective in the infrared spectrum and appears white on the camera image. *Therefore, it is not enough to create a black and white marker, then visually inspect it and assume that it works!*

9 Set up the application computer

The application computer is an external computer that runs touch-enabled applications. These can include Cornerstone-based applications, applications that support Windows Touch, TUIO-based applications, and applications that can receive tracking data in XML format (see section 9.1.

The application computer receives tracking data from the tracking computer, and sends video data back to the Cell for display on the LCD screen. Application computers can support Windows, Linux or OS X operating systems.

After the tracking engine has been set up (see section 8), you must configure the application computer to receive tracking data over the local network from the tracking engine:

1. First, you must install *touch receiver software* ie, the MultiTaction Cornerstone software package.
2. You will also need to specify the *display topology* ie, provide the application computer with details about the number, size and orientation of your Cells
3. Then you must set up Cornerstone to receive tracking data.
4. If required, you can also enable support for Windows multi-touch functionality ('Windows Touch'), TUIO applications, and tracking data received in XML format.

These tasks are described in the following sections.

Note: *Cable connections to the application computer are described in section 6*.

9.1 Tracking data transmission formats

Each touch-enabled application requires tracking data to be transmitted in a specific format. The following methods can be used to transmit tracking data:

- **Cornerstone:** MultiTaction Cells send tracking data in a proprietary format that can be interpreted by Cornerstone applications. To run Cornerstone applications on your Cells, you must install Cornerstone on the application computer. Cornerstone uses a binary data format. For more information, see section 9.3 and section 9.4.
- **Windows Touch:** This refers to the native multi-touch functionality in the Windows operating system. If you want to run Windows Touch applications, you must install Cornerstone on the application computer. Cornerstone includes a native Windows multi-touch driver to support Windows Touch. For more information, see section 9.9.
- **TUIO:** This refers to the TUIO 1.1 multi-touch protocol and API. If you want to run TUIO-based applications, MultiTaction Cells can send tracking data as TUIO streams. TUIO uses a binary data format. For more information, see section 9.10.
- **XML:** MultiTaction Cells can send tracking data in an XML stream to an XML stream server. Your applications can then connect to this server. XML tracking data is a text-based format, and so has a higher processing overhead than Cornerstone and TUIO formats. For more information, see section 9.11.

9.2 Supported operating systems

The latest release of Cornerstone (version 2.1) currently supports the following operating systems on an external application computer:

- **Windows:** Windows 7 or later
- **Linux:** Ubuntu 14.04 only. See also section 9.2.1.
- **OS X:** 10.10 Yosemite only

9.2.1 Install the base Linux operating system

Applies only to Linux application computers.

As a first step, you will need to install the base Linux operating system. You can find instructions and OS images at www.ubuntu.com. The supported Linux distribution is listed above.

9.3 Download the Cornerstone software

If you intend to use MultiTouch Cornerstone or need any of the utilities included with Cornerstone, you must install it on your external application computer. Cornerstone is available in two versions:

- **Runtime package:** This version is for end-users. It is used for most installations.
- **Software Development Kit (SDK):** This version is used to create new applications. The SDK package also includes the runtime components. If you install the SDK, you do not also need to install the runtime package.

Cornerstone software is available for the major 64-bit platforms ie, Windows, OS X and Linux. You can download it from:
https://cornerstone.multitouch.fi.

To run Cornerstone software on an application computer, you only need to download the runtime package for your OS.

Note: *You must be registered to download the software. Registration is free. You can create an account here:* https://cornerstone.multitouch.fi/user/register

9.4 Install the Cornerstone software

This section includes information for installing Cornerstone on an external application computer (Windows, Linux or OS X).

9.4.1 Install Cornerstone on a Windows application computer

Install the Cornerstone SDK or Cornerstone runtime. These are available as Windows installation packages.

- **Runtime:** This package names use this format:
  cornerstone-runtime-<version>-<timestamp>-<identifier>.exe

  For example:
  cornerstone-runtime-2.1.0-64bit-nightly-20151118-4fed225.exe
- **SDK:** This package includes full runtime binaries, so you do not need to install both the SDK and runtime packages on the same application computer.

  The SDK package names use this format:
  cornerstone-sdk-<version>-<timestamp>-<identifier>.exe

  For example:
  cornerstone-sdk-2.1.0-64bit-nightly-20151118-4fed225.exe
- **Config.txt and screen.xml:** These files are used to connect the application computer to the tracking computer and configure the Cell screens. The files are saved in your user profile directory.

  For example, if Cornerstone was installed by user Joe, then config.txt and screen.xml are saved here:
  C:\Users\Joe\AppData\Roaming\MultiTouch

9.4.2 Install Cornerstone on a Linux application computer

Install the Cornerstone SDK or Cornerstone runtime. These are available as Linux installation packages. These packages are shipped as executable shell scripts that install all dependencies and the relevant version of Cornerstone.

Note: *From Cornerstone 2.0 onwards, you can install multiple versions of Cornerstone on the same computer without library conflicts. For example, you can have one application using Cornerstone 2.0.0 while another uses Cornerstone 2.0.2.*

- **Runtime:** This package includes the Cornerstone libraries and applications as pre-compiled binaries. In addition, the script installs dependency packages such as multitouch-libav and multitouch-argyll.

  The runtime package names use this format:
  cornerstone_<version>_<Linux ver>_x86_<identifier>.sh.

  For example:
  cornerstone_2.0.8_Ubuntu-12.04_x86_64-ea8fa4f.sh

- **SDK:** This package includes full runtime binaries, so you do not need to install both the SDK and runtime packages on the same application computer. The package also contains an additional cornerstone-dev package that contains relevant header files and API documentation.

  The SDK package names use this format:
  Cornerstone-sdk-<version>_<Linux ver>_x86_<identifier>.sh.

  For example:
  cornerstone-sdk-2.0.8_Ubuntu-12.04_x86_64-ea8fa4f.sh

- **Config.txt and screen.xml:** These files are used to connect the application computer to the tracking engine and configure the Cell screen. These files are saved in the ~user/.MultiTouch subdirectory:

  For example, if Cornerstone was installed by user Joe, then config.txt and screen.xml are saved here:
  /home/joe/.MultiTouch

9.4.3 Install Cornerstone on an OS X application computer

Install the Cornerstone SDK or Cornerstone runtime. These are available as OS X packages.

- **Runtime:** This package names use this format:
  cornerstone-runtime-<version>-<OSX ver>-<timestamp>-<identifier>.pkg

  For example:
  cornerstone-runtime-2.1.0-OSX-10.10.5-20151118-4fed225.pkg

- **SDK:** This package includes full runtime binaries, so you do not need to install both the SDK and runtime packages on the same application computer.

  The SDK package names use this format:
  cornerstone-sdk-<version>-<OSX ver>-<timestamp>-<identifier>.pkg

  For example:
  cornerstone-sdk-2.1.0-OSX-10.10.5-20151118-4fed225.pkg

- **Config.txt and screen.xml:** These files are used to connect the application computer to the tracking engine and configure the Cell screen. These files are saved in your user library directory:

  For example, if Cornerstone was installed by user Joe, then config.txt and screen.xml are saved in this folder:
  /Users/Joe/Library/MultiTouch

9.5 Specify the Cell IP addresses

In order for touch-enabled applications to receive tracking data, the application computer needs to know the network address of each Cell's tracking engine. To identify the tracking engine, you must edit config.txt.

To specify the network address of the tracking engine:

1. Make a note of each Cell's IP address. This address is shown in the lower left corner of the OSD eg, 10.0.0.46. See section 8.3.1
2. Locate config.txt on the application computer. The file location depends on the operating system:

   Windows: see section 9.4.1.

   Linux: see section 9.4.2.

   OS X: see section 9.4.3.
3. Open config.txt in a text editor.
4. Go to the `NetBridge` block and change the `host` value to match the IP address of the tracking engine on the first Cell. For example:

```
NetBridge {
     host = "10.0.0.46"
}
```

5. (*Applies only if you have multiple Cells*) Add a `NetBridge` block for each additional Cell and repeat step 4. In the example below, config.txt identifies three Cells:

```
NetBridge {
  host = "10.0.0.46"
}
NetBridge {
  host = "10.0.0.47"
}
NetBridge {
  host = "10.0.0.48"
}
```

9.6 Specify the display topology

In order to display applications correctly on your display or video wall, you need to provide the application computer with details about the Cell *topology* ie, the number, size and orientation of your Cells.

This subject is covered separately in section 10.

9.7 Set up Cornerstone applications

Cornerstone includes a set of demo applications for calibrating and testing the displays. It also includes a set of customizable stock applications.

Note: Documentation for Cornerstone applications can be found here:
https://cornerstone.multitouch.fi/application

9.7.1 Demo application: MultiTouch Bench

Bench is a simple application that lets you interact with the basic elements of Cornerstone applications.

To start Bench, double-click the Bench icon in the file manager or run the following commands:

- **Windows**
  ```
  cd C:\\Cornerstone-x.y.z\\bin
  MultiTouchBench.exe
  ```
- **Linux**
  ```
  /opt/cornerstone-x.y.z/bin/MultiTouchBench
  ```
- **OS X**
  ```
  /opt/cornerstone-x.y.z/bin
      /MultiTouchBench.app/Contents/MacOS/MultiTouchBench
  ```

Where `x.y.z` refers to your Cornerstone version such as `2.0.0`.

9.7.2 Demo application: Twinkle

Twinkle is a simple drawing application. It demonstrates basic interaction and how to acquire the input from the touchscreen. The application has different styles for the graphics that can be switched with the 's' key. Clear the screen with the 'z' key or tapping the "CLEAR" text in the upper left corner.

To start Twinkle, double-click the Twinkle icon in the file manager or run the following commands:

- **Windows**
  ```
  cd C:\\Cornerstone-x.y.z\\bin
  Twinkle.exe
  ```
- **Linux**
  ```
  /opt/cornerstone-x.y.z/bin/Twinkle
  ```
- **OS X**
  ```
  /opt/cornerstone-x.y.z/bin/Twinkle.app/Contents/MacOS/Twinkle
  ```

Where `x.y.z` refers to your Cornerstone version such as `2.0.0`.

9.8 Cornerstone updates

To update Cornerstone, you must download and install a new runtime or SDK package. You do not need to uninstall the existing Cornerstone package. You can have multiple versions of the SDK installed at the same time; you just need to specify a different installation directory for each version.

If you prefer to clean up the old Cornerstone installation (for example, to save disk space), do the following:

- **Windows:** Run the Cornerstone uninstaller from the installation folder, or use the Control Panel to uninstall Cornerstone.
- **Linux:** Uninstall the packages using:
  `apt-get remove cornerstone-2.1.[package]`
- **OS X:** Apple package management does not support uninstallation. This is a characteristic of Apple software, so there is no simple way to uninstall software. Instead, you can manually remove the following directories:
  `/opt/multitouch`
  `/opt/cornerstone-VERSION`.

9.9 Enable support for Windows Touch functionality

Cornerstone includes support for the native Windows multi-touch functionality. A native Windows multi-touch driver (the 'MultiTouch Cell' driver) is included in both the runtime and SDK packages, available from download area. These instructions describe how to install support for Windows multi-touch functionality on an external computer.

9.9.1 Terminology

The terminology for this area can be confusing. In Windows 7, support for single touch functionality was officially named *Windows Touch*. From Windows 8 onwards, this functionality was renamed to *Windows Pointer Device* and extended to include both single touch and multi-touch functionality.

From the Microsoft MSDN web site:

> "Windows Touch is the name for the touch and multi-touch functionality in the Windows 7 operating system. In the context of Windows Touch, touch refers to support of a single physical contact point, whereas multi-touch refers to support for two or more concurrent physical contacts." [1]

See also:

> "Starting in Windows 8, Touch has been expanded to include any pointer device and is now referred to as Windows Pointer Device. Windows Pointer Device refers to devices that support pen (stylus input), touch, or multi-touch functionality."[2]

[1] https://msdn.microsoft.com/en-us/library/windows/hardware/ff553747(v=vs.85).aspx

[2] https://msdn.microsoft.com/en-gb/library/windows/hardware/hh406720(v=vs.85).aspx

9.9.2 Install the native Windows multi-touch driver

On the application computer:

1. Confirm that the Cornerstone runtime or SDK is already installed; see section 9.4.
2. Browse to the folder C:\Cornerstone-VERSION\Win7Driver.
   a. Right click the install.bat file and select 'Run as administrator'.
   b. After the driver has installed, reboot the application computer.

   Note: *Although the folder name is Win7Driver, the driver also works on Windows 8 and Windows 10 systems.*
3. When the application computer restarts, start the MTWin7.exe program included with the runtime. While MTWin7.exe is running, any application that supports Windows Touch can use the tracking data received by Cornerstone.

   Note: *We recommend you add MTWin7.exe to the list of Startup programs on the application computer. This ensures that MTWin7.exe is always running when it is needed by an application that supports Windows Touch.*

9.9.3 Uninstall the native Windows multi-touch driver

On the application computer:

1. Open the Device Manager
2. Locate the MultiTouch Cell driver, listed under Human Interface Devices.
3. Right-click the driver and select Uninstall.
4. Select the check box "Delete driver software for this device" and click OK.

9.9.4 Native Windows multi-touch driver notes

- *The native Windows driver currently only works with a single Cell. This is due to limitations in the Windows Touch interface, which only supports a single touch display per Windows computer. See section 14.2.2.*
- *Most Windows applications do not support multi-touch, especially if there are multiple users at the same time. However, Microsoft Paint does support multi-touch input and can be used to test that the driver has been setup correctly.*

9.10 Enable support for TUIO-based applications

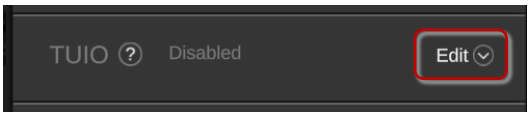
"TUIO is an open framework that defines a common protocol and API for tangible multitouch surfaces." www.tuio.org

The TUIO protocol can be used to transmit touch events (ie, tracking data) from MultiTaction Cells to any client application that supports the protocol. If you want to run TUIO-based applications, you must enable TUIO transmission on the Cell.

9.10.1 Enable TUIO transmission

TUIO transmission is disabled by default on MultiTaction Cells. To enable TUIO transmission:

1. Display the OSD and tap the Setup tab.
2. Go to the TUIO section and tap the Edit button to display the TUIO settings.



OSD Setup tab, TUIO section, Edit button.

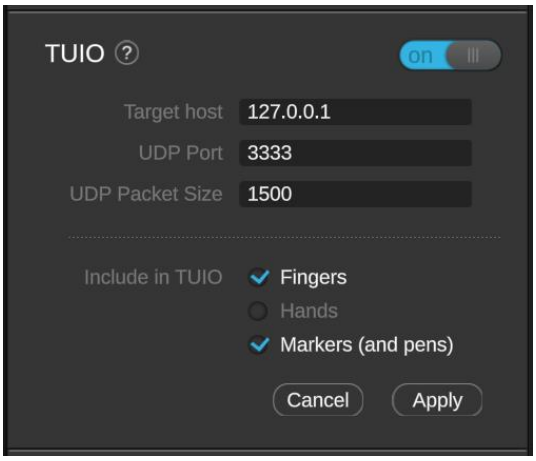
3. Configure the settings as required:

   Target host: Enter the IP address of your application computer.

   UDP Port: This port is typically 3333, but may vary depending on the client application.

   UDP Packet Size: A typical Ethernet frame size is 1500 bytes. You rarely need to adjust the default packet size.

   Include in TUIO: Specify which types of tracking data (for example, markers and infrared pens) are included in TUIO transmissions.



OSD Setup tab, TUIO transmission enabled for finger and marker tracking data.

4. Click Apply.
5. Tap the Save button to save the TUIO settings.

   You may need to scroll to the bottom of the Setup tab to see the Save button.

9.10.2 TUIO transmissions from multiple Cells

An individual Cell can directly provide TUIO output that matches a single screen. But if you have a display comprising multiple Cells, this direct approach does not work automatically because individual Cells are not aware of other Cells.

To use TUIO with multiple MultiTaction Cells, you must first create a tracking hub on the application computer. The tracking hub collects tracking data from multiple Cells and sends out the data as normalized TUIO coordinates in a single coordinate stream.

Set up a tracking hub

1. Create a config.txt file for multiple MultiTaction Cells; see section 9.5, step 5.
2. Add a `TUIOSender` block to config.txt.

   The `TUIOSender` block instructs the Cornerstone application to send the tracking data to specified host ie, the application computer. For example:

   ```
   TUIOSender {
       features = "fingers objects"
       address = "localhost"
       port = "3333"
   }
   ```

   This example sends finger tracking data to the local application computer. For full config.txt syntax, see appendix C.1.
3. Run a Cornerstone application to serve as a tracking hub, collating tracking data from multiple Cells and sending the data to the client application as a single TUIO stream.

   Any Cornerstone application can act as a tracking hub. The most popular applications are *HelloWorld* (for simple visual feedback) and *MTServer* (for operating without any windows).

Example config.txt

Using the example two Cell setup from section 10.5.2, the complete config.txt file would include the following `TUIOSender` block and two `NetBridge` blocks:

```
TUIOSender {
    features = "fingers objects"
    address = "localhost"
    port = "3333"
}
NetBridge {
    host = "10.0.0.1"
    input-translate = "0 0"
}
NetBridge {
    host = "10.0.0.2"
    input-translate = "1930 0"
}
```

The `NetBridge` and `TUIOSender` blocks are independent of each other:

1. There are two `NetBridge` blocks, one for each Cell. Each `NetBridge` block specifies a Cell IP address and how to transform tracking data sent from that Cell.
2. The Cornerstone tracking hub combines the tracking data from the two Cells into a single tracking stream.
3. The `TUIOSender` block instructs the tracking hub to send the single stream of tracking data as a TUIO stream to the receiving application, identified by its host IP address and port.

9.11 Enable applications to receive XML tracking data

Applications can also receive tracking data in XML format. You may want to do this if you are not using Cornerstone- or TUIO-based applications.

All MultiTaction Cells send tracking data in an XML stream. You can set up a Cornerstone application to send this XML tracking data to an XML stream server. Your applications can then connect to this server.

9.11.1 Set up an XML stream server

1. Create a config.txt file for multiple MultiTaction Cells; see section 9.5, step 5.
2. Add a `Globals` block to config.txt.

   The `Globals` block instructs the Cornerstone application to send XML tracking data to the XML server socket, identified by its hostname and port numbers. For example:

   ```
   Globals {
       xml-handshake-port = "5503"
       xml-server-host = "localhost"
       xml-server-port = "5501"
   }
   ```

   Here, the hostname of the XML server socket is 'localhost'. Default port numbers for the XML handshake server socket and XML server socket are 5503 and 5501 respectively. For full config.txt syntax, see appendix C.1.
3. Run a Cornerstone application. This application will serve as an XML stream server.

   Any Cornerstone application is capable of acting as an XML stream server. Usually, *MTServer* is used for this purpose.

9.11.2 Example config.txt

Using the example two Cell setup from section 10.5.2, the complete config.txt file would include the following `Globals` block and two `NetBridge` blocks:

```
Globals {
    xml-handshake-port = "5503"
    xml-server-host = "localhost"
    xml-server-port = "5501"
}
NetBridge {
    host = "10.0.0.1"
    input-translate = "0 0"
}
NetBridge {
    host = "10.0.0.2"
    input-translate = "1930 0"
}
```

In this example:

1. There are two `NetBridge` blocks, one for each Cell. Each `NetBridge` block specifies a Cell IP address and how to transform tracking data sent from that Cell.
2. The `Globals` blocks instructs the Cornerstone application to send XML tracking data from the two Cells to the specified XML stream server socket, identified above as 'localhost', using the default socket port numbers.
3. Your application can now open a socket to the XML stream server (using port 5501) and start receiving tracking data in XML form.

10 Configure the display

You now need to provide the operating system and your touch-enabled applications with details about the *display topology* ie, the number and orientation of your Cells and how they are arranged in relation to each other.

Before you configure the display, it is helpful to understand how the Cell graphics system works. In particular, it is important to understand the screen coordinates used by the graphics system. For full details, see section 10.2.

10.1 Procedure summary

Configuring a display topology is a three-step procedure:

1. Provide the operating system with details about the position and orientation of your Cells.

   To do this, you typically use a proprietary NVIDIA or Windows configuration tool to configure the driver for the Cells graphics card(s).

   For full details, see section 10.2.

2. Provide your touch-enabled applications such as Cornerstone with details about where they get displayed on the Cell screens.

   To do this, you edit the screen.xml configuration file. Screen.xml maps an application's *logical* screen area onto the Cells' *physical* screen area.

   For details, see section 10.3.2.

3. Provide Cornerstone with information about the number and orientation of Cells so it can correctly interpret touch coordinates in the tracking data that it receives.

   Specifically, Cornerstone must be able to combine streams of tracking data from multiple Cells into a single coordinate system that matches the display topology. Also, Cornerstone assumes that Cells are in landscape mode. If Cells are installed in portrait mode, Cornerstone needs to transform the tracking data coordinates accordingly.

   To combine and transform the tracking data received by Cornerstone, you must edit config.txt. For details, see section 10.5.

Note that a key task in steps 1 and 2 is to add *bezel compensation*. For an explanation of bezel compensation, see section 10.1.1.

Finally, most Cell deployments in the real world are fairly simple, with the application computer configured to display applications on a single desktop that extends across the Cells' entire screen surface. However, the Cell graphics system is highly flexible, allowing it to support complex display topologies; for a brief explanation, see section 10.1.2.

10.1.1 What is bezel compensation?

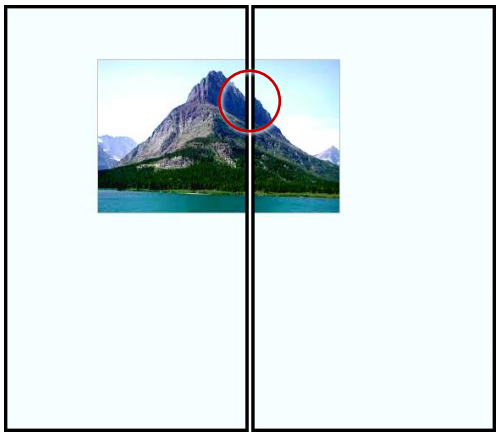
A *bezel* is the protective external rim around the front glass panel of a Cell. When two Cells are positioned next to each other with no space between the units, the two adjacent bezels have a combined width of approximately 5mm.

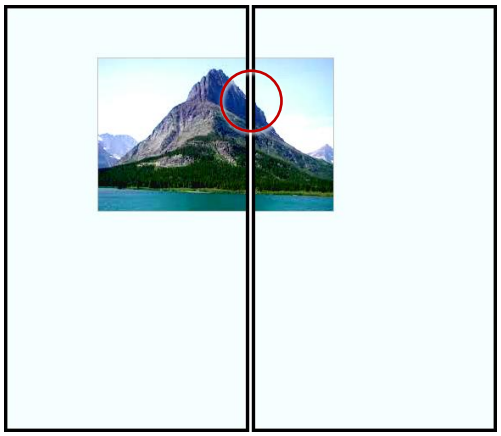
If you have two or more Cells in your display, you need to configure the operating system and your touch-enabled applications to compensate for the bezel between Cells. Even though the bezel is very thin, any images that straddle a bezel will look offset if the display coordinates are not adjusted to compensate for the bezel.

Consider a photo of a mountain that straddles two Cells. The Cells form a single continuous display, but even when the Cells are physically adjacent their bezels mean there is a 5mm vertical gap between the LCD screens that cannot display the photo.

The GPU is unaware of this gap. Without bezel compensation (**1**), the entire photo is displayed but the part in the right-hand Cell is a direct continuation from the left-hand Cell. It is offset and looks unnatural.

When the GPU is configured to include bezel compensation (**2**), the photo extends into the right-hand Cell as though the bezel were part of the LCD display. A 5mm vertical strip of photo is now hidden, but there is no offset and the overall effect looks more natural. To the user, it appears as though the photo extends behind the bezel.





1 *No bezel compensation*

2 *With bezel compensation*

(Note that the difference is subtle in the diagrams above. The effect is more noticeable when viewed on full size screens.)

10.1.2 Complex display configurations for application developers

In the real world, most display configurations are simple. For example, the MultiTaction Meeting Room solution comprises three adjoining Cells in portrait mode. MultiTaction provide predefined configuration files that enable the operating system and touch-enabled applications to recognize this layout and compensate for the thin bezel between the Cells.

However, you may need to define a more complex display configuration if:

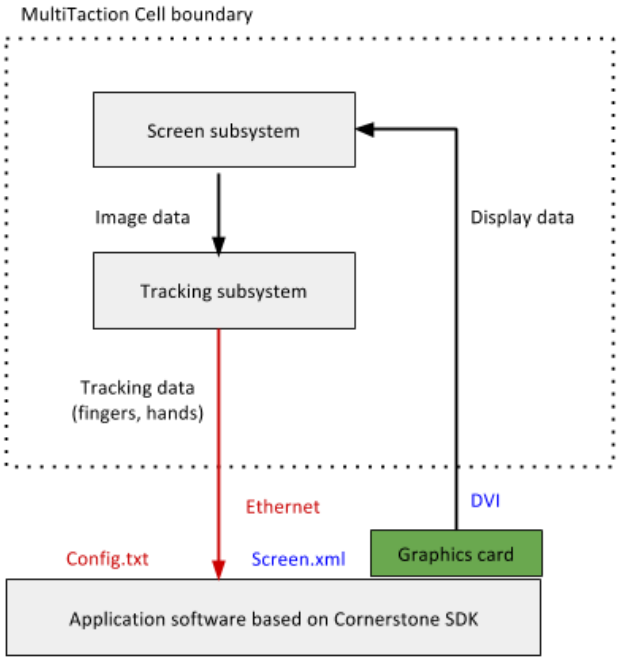
- You want a non-standard display topology.
- You are developing Cornerstone applications to run on a multiple-Cell display but you do not have the physical Cells available to test the application. The flexibility in the Cell graphics display system allows you to code the application to simulate performance on a multiple Cell display even if, for example, you only have access to a single screen.

10.2 Cell graphics system

Logically, a Cell consists of a *screen subsystem* and a *tracking subsystem*:

1. The screen subsystem generates a visible display that extends across the Cells' entire screen surface. It also captures images of users' hands and other pointer devices when the Cell screen is touched.
2. The image data is fed to the tracking subsystem, more commonly called the *tracking engine*; see section 9. The tracking engine runs on the Cell's internal computer and uses computer vision algorithms to recognize the location of users' fingers and other pointer devices and outputs this information as tracking data. The amount of tracking data is much less than the image data.

The screen subsystem and tracking engine are physically located inside the Cell.



Cell graphics system, with screen and tracking subsystems.
Note that the diagram mixes software and hardware components.

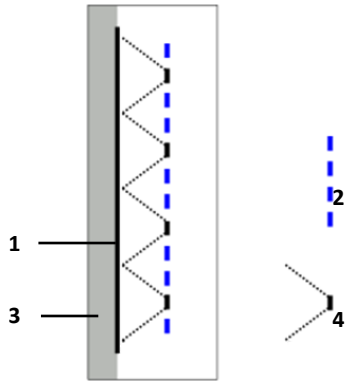
10.2.1 Hardware schematic

The diagram below shows a simplified hardware schematic of a Cell's graphics system for MT420* and MT550* models.

Briefly, an LCD panel (**1**) generates the visible image by filtering the white light emitted by the background LEDs (**2**) on a pixel by pixel basis. Infrared LEDs (**2**) emit non-visible light *through* the LCD panel and front glass (**3**).

An array of infrared cameras (**4**) are aimed at the LCD panel. These cameras capture frames at a high frame rate and send image data to a computing unit.

Users' fingers, hands and markers on the front glass reflect the infrared light and are recognized using machine vision algorithms in the computer. Likewise, if a user touches the screen with an infrared pen, the machine vision algorithms recognize the pen.



*Graphics system hardware. **1** LCD display. **2** Background LEDs emitting white light and infrared LEDs emitting non-visible light. **3** Front glass. **4** Infrared cameras aimed at the LCD display.*

10.2.2 Graphics coordinate systems

The *drawable region* extends across the Cells' screen surface and represents the desktop on which touch-enabled applications get displayed. Within the drawable region, you need to specify a *physical screen area* available for displaying applications, plus an application's *logical screen area*. You define these concepts in the screen.xml configuration file.

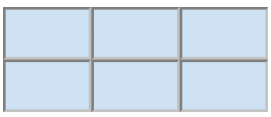
Terminology

The graphics coordinate system uses the following terminology in screen.xml:

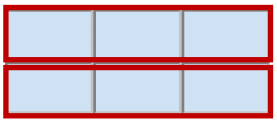
`window` The `window` element in screen.xml defines a drawable region available for displaying applications. It extends across the Cells' screen surface and is similar to a computer desktop. It is also sometimes called the *operating system window*. You must specify the `window` location and size.

For efficient rendering, we recommend that you define one `window` per GPU on the application computer. For example:

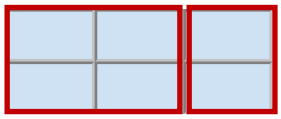
- If you have four Cells in your display, driven by a single quad head GPU, you only need one `window` extending across all four screens.
- If you have 6 Cells driven by two quad head GPUs, you need two `windows`. You could have both `windows` extending across three screens. Or you could specify one `window` extending across four screens and the second `window` extending across two screens:



Six Cell display driven by two GPUs



GPU 1: Window 1, 3 Cells
GPU 2: Window 2, 3 Cells



GPU 1: Window 1, four Cells
GPU 2: Window 2, two Cells

`area` The `area` element represents the *physical* screen area where applications are displayed. An `area` has a fixed size and location and is measured in pixels. (In practice, an `area` is an OpenGL viewport.)

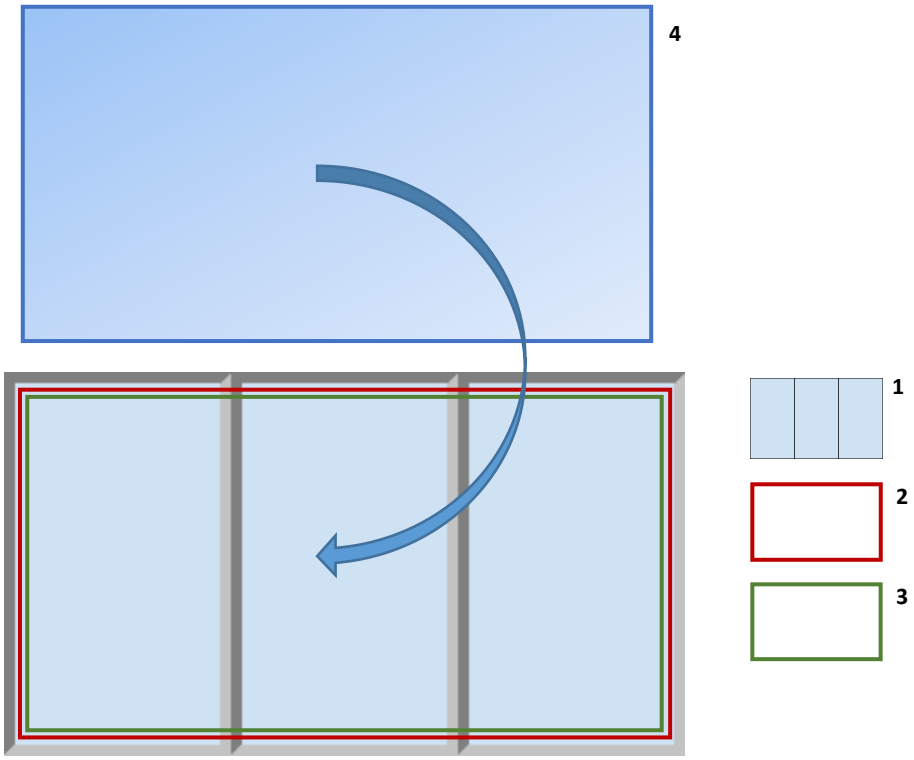
For efficient rendering, we recommend a single `area` for each `window`. In a simple display topology, one `area` maps exactly onto one `window`.

`graphics` The `graphicssize` and `graphicslocation` elements together define a *logical* screen area for displaying an application's graphics. For each MultiTouch display, there is only one logical screen area that extends across all available physical screen areas.

Using the data in screen.xml, a Cell will automatically map an application's logical screen area onto its physical screen area. If necessary, it can scale the logical screen area to fit the available physical screen area.

The `graphics` elements can also be used to add bezel compensation. See page 61 for an example.

The `window`, `area` and `graphics` elements are illustrated on page 60.



Example graphics coordinate system

1 *Entire screen area, comprising three Cells.*

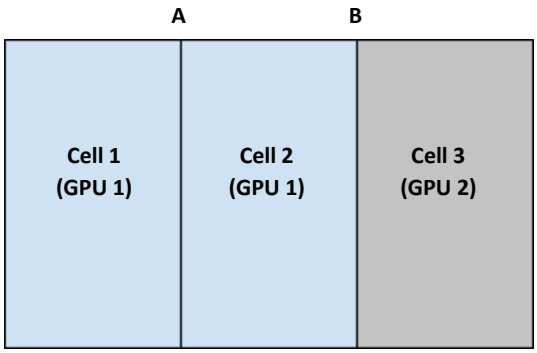
2 *Drawable region spans all Cells driven by one GPU. Defined by* `window` *element. If a display requires multiple GPUs, you will need multiple drawable regions.*

3 *Physical screen area for applications. Defined by* `area` *element. You typically need one physical screen area for each drawable region.*

4 *Logical screen area for applications. Defined by* `graphics` *elements. The application computer maps this logical screen area onto the available physical screen area, scaling or resizing if necessary* (**3**).

10.2.3 Bezel compensation example: vertical bezel

Consider a Meeting Room display with three vertical 1920 x 1080 Cells. For this example, the display is driven by two dual-head GPUs. One GPU drives Cells 1 and 2; a second GPU drives Cell 3. There are two vertical bezels A and B.



Meeting Room display

GPU 1 drives window 1, extending over Cells 1 and 2.

GPU 2 drives window 2, extending over Cell 3.

A *Bezel between Cells 1 and 2*

B *Bezel between Cells 2 and 3*

To compensate for bezel A between Cell 1 and Cell 2, you can simply configure the GPU driver (see section 10.3). But to compensate for bezel B between Cells 2 and 3, you must edit screen.xml. This is because GPU 2 is unaware of the gap between Cells 2 and 3.

1. Define a window, physical screen area and logical screen area extending across Cell 1 and Cell 2. Pay particular attention to the *size* of each element:
   - Set the window *size* to 2170 x 1920 pixels, where the 2170 width is two Cell widths plus bezel A ie 1080 + 10 + 1080 pixels.
   - Also set the *size* of the physical screen area to 2170 x 1920 pixels.
   - Also set the *size* of the logical screen area to 2170 x 1920 pixels.

   For each element, its *location* is the standard 0,0.
2. Define a second window plus a physical screen area and logical screen area for Cell 3. Pay particular attention to the *location* of each element:
   - Shift the window *location* to 2170,0. The 2170 x-coordinate locates the second window immediately right of the first window.
   - Set the *location* of the physical screen area to 0,0 ie, it matches exactly the location of the second window.
   - Shift the *location* of the logical screen area to 2180, 1920. The 2180 x-coordinate shifts the second logical screen area to the right by 10 pixels of the first logical screen area plus bezel B ie, 2170 + 10 pixels.

   For each element, its *size* is 1080 x 1920 ie, the standard size for a single Cell.

   A 6mm vertical strip of application graphics is now hidden by bezel B, but the overall visual effect will look more natural.

The required changes to screen.xml for this example are shown on page 62.

Here is compensation for bezels A and B as it would appear in screen.xml.

```
<!DOCTYPE mtdoc>
<MultiHead type="">

  <GPU_1 type="window">
    <location type="">0 0</location>
    <size type="">2170 1920</size>
                /* window width 2170 = 1080+10+1080 */
    <Area1 type="area">
      <location type="">0 0</location>
      <size type="">2170 1920</size>
                /* physical screen area width 2170 = 1080+10+1080 */
      <graphicslocation type="">0 0</graphicslocation>
      <graphicssize type="">2170 1920</graphicssize>
                 /* logical screen area width 2170 = 1080+10+1080 */
    </Area1>
  </GPU_1>

  <GPU_2 type="window">
    <location type="">2170 0</location>
                 /* window location 2170 = 1080+10+1080 */
    <size type="">1080 1920</size>
    <Area1 type="area">
      <location type="">0 0</location>
      <size type="">1080 1920</size>
      <graphicslocation type="">2280 0</graphicslocation>
                 /* logical screen area location is 2180,0 where */
                 /* 2180 = 1080+10+1080+10                       */
      <graphicssize type="">1080 1920</graphicssize>
    </Area1>
  </GPU_2>

</MultiHead>
```

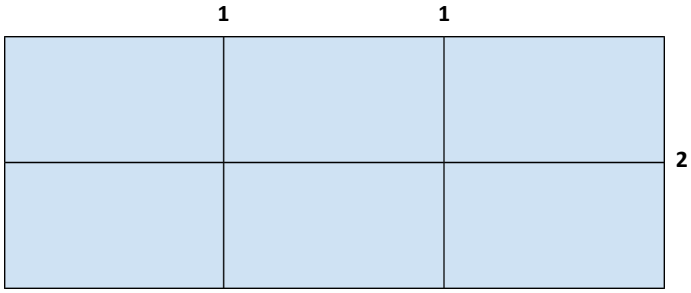
Screen.xml with bezel compensation.

***Size** changes in the first window add compensation for bezel A.*

***Location** changes in the second window add compensation for bezel B.*

10.2.4 Bezel compensation example: horizontal bezel

Consider a 'Board Room' display comprising two rows of three 1920 x 1080 Cells, driven by two quad-head GPUs. One GPU drives the three Cells in the top row; a second GPU drives the three Cells in the bottom row. There are two vertical bezels and one horizontal bezel.



Board Room display. **1** *Vertical 5mm bezel.* **2** *Horizontal 5mm bezel.*

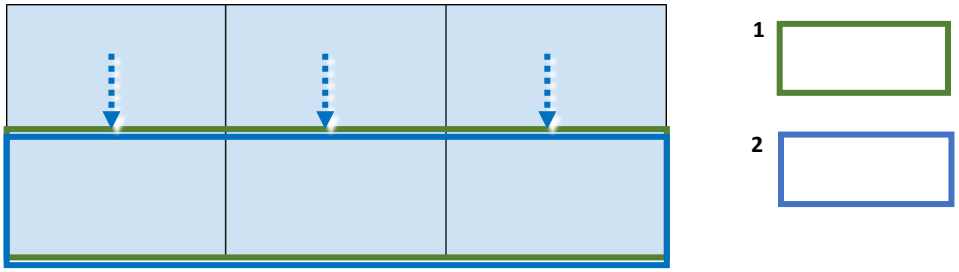
To compensate for the vertical bezels, you can configure the GPU driver (see section 10.3), but to compensate for the horizontal bezel you must edit screen.xml:

3. Define a `window` and physical screen `area` extending across the top row.
4. Define a logical screen area for the top row, using `graphicslocation` and `graphicssize`, to exactly match the top `area` size and location
5. Define a second `window` and physical screen `area` extending across the bottom row. Each is located immediately below the corresponding `window` and `area` in the top row.
6. Define a second logical screen area for the bottom row, but this time the logical screen area starts 6mm *below* the physical screen area in the bottom row.

   Now, `graphicssize` exactly matches the size of the bottom row `area` but the `graphicslocation` is offset by (0, 10), where 10 pixels roughly equals 6mm.

   A 6mm horizontal strip of application graphics is now hidden by the bezel, but the overall visual effect will look more natural.

The required changes to screen.xml for this example are shown on page 64.



Board Room display. **1** *Physical screen area defined for bottom row of Cells.* **2** *Logical screen area for bottom row of Cells, shifted downwards 6mm to compensate for horizontal bezel.*

Here is the horizontal bezel compensation as it would appear in screen.xml.

```
<!DOCTYPE mtdoc>
<MultiHead type="">

  <GPU_1 type="window">                 /* Top row window                      */
    <location type="">0 0</location>    /* Top row window location             */
    <size type="">5780 1080</size>      /* Top row window size                 */
    <Area1 type="area">                 /* Top row area                        */
      <location type="">0 0</location>  /* Top row physical screen location    */
      <size type="">5780 1080</size>    /* Top row physical screen size        */
      <graphicslocation type="">0 0</graphicslocation>
                                        /* Top row logical screen location     */
      <graphicssize type="">5780 1080</graphicssize>
                                        /* Top row logical screen size         */
    </Area1>
  </GPU_1>

  <GPU_2 type="window">                 /* Bottom row window                   */
    <location type="">0 1080</location> /* Bottom row window location          */
    <size type="">5780 1080</size>      /* Bottom row window size              */
    <Area1 type="area">                 /* Bottom row area                     */
      <location type="">0 1080</location>
                                        /* Bottom row physical screen location */
      <size type="">5780 1080</size>    /* Bottom row physical screen size     */
      <graphicslocation type="">0 1090</graphicslocation>
                                        /* Bottom row logical screen location  */
                                        /* offset vertically by +10 from       */
                                        /* physical location ie, 1080+10=1090  */
      <graphicssize type="">5780 1080</graphicssize>
                                        /* Bottom row logical screen size      */
    </Area1>
  </GPU_2>

</MultiHead>
```

Screen.xml with horizontal bezel compensation added by the `<graphicslocation>` element in the GPU_2 window.

Note that logical screen area for the `GPU_2` window (the bottom row) is offset by +10 to compensate for the horizontal bezel between the top and bottom rows of Cells. Any images that straddle the top and bottom rows will now appear to users as though they continue behind the bezel. For example, if you slowly drag a photo down from the top row to the bottom row of Cells, its bottom edge briefly 'disappears' behind the bezel before re-emerging on the lower screens.

Without this bezel compensation, any images that straddle the top and bottom rows would appear to 'jump' between rows. For example, if you slowly dragged a photo down from the top row to the bottom row of Cells, its bottom edge will immediately jump from the upper screens to the lower screens.

10.3 Configure the operating system

When configuring the display topology, the first step is to configure the operating system on the application computer. The operating system needs to recognize each connected individual Cell and to understand how the Cells are arranged.

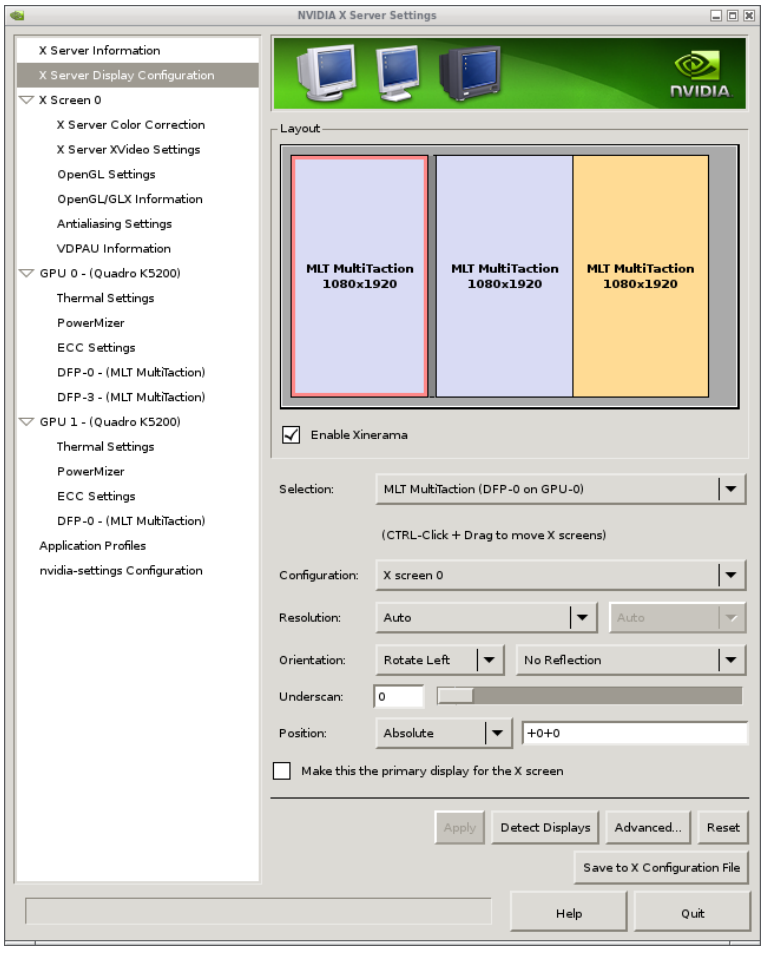
To provide the operating system with these details, you must configure the driver for the Cell graphics card(s). The driver runs on the application computer and automatically detects the number of Cells in your display. You provide the driver with details about Cell orientation and how Cells are arranged in relation to each other in your display.

Typically, you use a proprietary configuration tool to configure the driver. MultiTaction Cells use NVIDIA graphic cards, so you use the NVIDIA X Server Settings tool.

Note: *This step is not necessary if you do not intend to use the desktop of the application computer. (That is, the application computer will only run Cornerstone applications during the Cell installation stage.) In these circumstances, you do not need to provide the operating system with details about the correct display setup. This is because Cornerstone can use setup details defined in its own configuration files to compensate for any setup defined in the operating system.*

10.3.1 NVIDIA configuration tool: X Server Settings

On application computers with an NVIDIA GPU, use the X Server Settings tool to specify the layout of your Cells (order and rotation). For usage instructions, please refer to your NVIDIA documentation.



NVIDIA X Server Settings dialog, Display Configuration screen

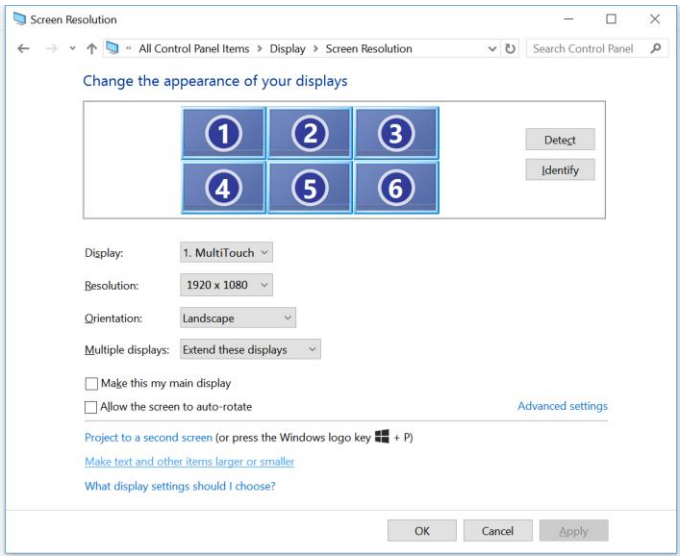
10.3.2 Windows configuration tool: Display applet

On a Windows application computer, you can use the Display applet as an alternative to the NVIDIA configuration tool. Be aware that the NVIDIA tool offers advanced configuration options (such as bezel compensation) that are not available in the Display applet.

To use the Windows Display applet:

1. After connecting your Cells and installing the latest GPU drivers, reboot the computer.
2. When the computer restarts, launch the Display applet in the Windows Control Panel.
3. Open the Screen Resolution dialog and drag the numbered displays to their correct locations that match the physical arrangement of your Cells.

Note: *In previous versions of Windows, the Display applet was variously called Display Properties or Display Settings. However, the configuration options available in the applet have generally remained constant over the Windows releases.*



Screen Resolution dialog, Windows 10.
Use the Identify button to see which physical Cell matches a numbered display

10.4 Specify where applications are displayed on the Cell screens

Now you need to specify where touch-enabled applications such as Cornerstone get displayed on the Cell screens. To do this, you edit the screen.xml configuration file. Screen.xml maps an application's *logical* screen area onto the Cells' *physical* screen area.

10.4.1 Procedure summary

1. Locate screen.xml on the application computer. The file location depends on the operating system:

   Windows: see section 9.4.1.

   Linux: see section 9.4.2.

   OS X: see section 9.4.3.

2. Edit screen.xml to include details about where applications display on the topology.

   Important! *Before you edit screen.xml, you need to understand the various graphics coordinates systems; see section 10.2.1.*

   The key XML elements that you need to edit are:

   - `<graphicslocation>` and `<graphicslocation>` in the `<area>` element. These define an application's logical display area.
   - `<location>` and `<size>` in the `<area>` element. These define the physical screen region on which an application can be displayed. To ensure efficient rendering, we recommend that you specify one `<area>` element that maps exactly onto the `<window>` element.
   - `<location>` and `<size>` in the `<window>` element. These define the operating system window ie, the entire screen area or desktop available to applications. A `<window>` can span multiple-Cell screens. You need one `<window>` for each GPU on the application computer(s). For example, if your display comprises 12 Cells powered by three quad-head GPUs, you need three `<window>` elements.

   The relationship of these elements within screen.xml is summarized below:

```
<window>
   <area>
      <graphicslocation>
      <graphicssize>
      <location>
      <size>
   </area>
   <location>
   <size >
</window>
```

   More information:

   - Example screen.xml setups are included in section 11.
   - Full syntax details for screen.xml are included in section C.2.

3. Test the display configuration.

   The easiest way to test the display configuration is to run MultiTouch Bench (see section 9.7.1) or some of the other Cornerstone template applications.

10.5 Combine and transform the tracking data

Cornerstone-based applications receive tracking data from the tracking engines running on each Cell connected to the application computer. Inside each Cell, the tracking engine comprises a set of *trackers* that exclusively process and transmit different types of tracking data (fingers, markers or infrared pens).

However, a tracker in one Cell is unaware of trackers in other Cells. Each Cell outputs its own stream of tracking data using coordinates relative to its own *origin* (ie, the upper left corner of its screen).

When Cornerstone receives a separate stream of tracking data from each Cell, it must be able to combine this data into a single coordinate system that matches the physical layout of the screens ie, the Cell topology. For an example, see section 10.5.2.

Likewise, Cornerstone by default assumes that Cells are in landscape mode. If your Cells are in portrait mode, Cornerstone needs to transform the tracking data coordinates accordingly. For an example, see section 10.5.3.

10.5.1 Procedure summary

To combine and transform the tracking data received by Cornerstone, you must edit config.txt.

1. Locate config.txt on the application computer. The file location depends on the operating system:

   Windows: See section 9.4.1.

   Linux: See section 9.4.2.

   OS X: See section 9.4.3.

2. Edit the relevant translation and rotation parameters in config.txt:

   - `input-translate` translates the tracking coordinates by an ‘x y’ amount, where ‘x y’ is the offset in pixels from the screen origin.
   - `input-rotate` gives the rotation of the Cell. Rotation is express in degrees, nominally from -180 to 180.

     Note: *In Cornerstone 1.2.3 and earlier, rotation was expressed in radians (nominally from -pi to pi).*

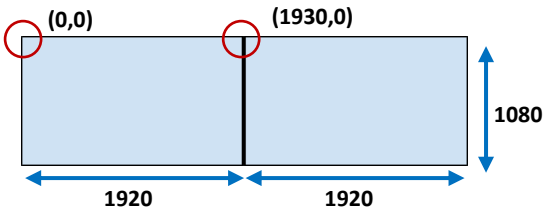
   For full syntax details, see section C.1.

3. Test the configuration.

   The easiest way to test the display configuration is to run MultiTouch Bench (see section 9.7.1) or some of the other Cornerstone template applications.

10.5.2 Example: Combine tracking data from two Cells

Two 1920 x 1080 Cells are in landscape mode:

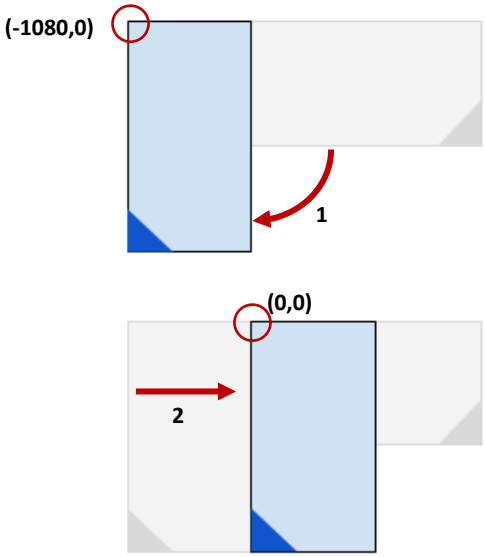


Each Cell outputs tracking data in its own coordinates. To combine and transform the tracking data into single coordinate system, add the following to config.txt. In this example, you only need to translate the tracking data from the right-hand Cell, shifting its x coordinate to the right by 1930 pixels (1920 pixel screen width plus 10 pixel bezel).

```
NetBridge {
  host = "10.0.0.1"
  input-translate = "0 0"
}
NetBridge {
  host = "10.0.0.2"
  input-translate = "1930 0"    /* x coordinate translated by +1920 */
}
```

10.5.3 Example: Rotate tracking data from a Cell in portrait mode

A single Cell is in portrait mode, with IP address 10.0.0.1. In this example, the Cell is rotated 90 degrees *clockwise*.



Cell rotated clockwise to portrait mode

1 *Cell rotated clockwise 90°. Coordinates effectively shifted left by 1080 pixels.*

2 *Coordinates translated right by 1080 pixels to compensate for left-shift in* 1.

Now you need to rotate and translate the tracking data coordinates. To make the tracking coordinate origin (0,0) match the upper left corner of the rotated screen, you must shift the x coordinate 1080 pixels to the right. Add the following blocks to config.txt:

```
NetBridge {
  host = "10.0.0.1"
  input-rotate = "90"          /* tracking data rotated by +90 degrees */
  input-translate = "1080 0"   /* x coordinate translated by +1080     */
}
```

Rotation around the upper left corner (**1**) is applied first, while translation to the right (**2**) is applied second. However, the order of `input-rotate` and `input-translate` in config.txt does not matter.

11 Example display configurations

This section includes example configurations in screen.xml and config.txt for the following display topologies.

- Single Cell in landscape mode: section 11.1
- Two Cells in landscape mode: section 11.2
- Six Cells in landscape mode ('Board Room'): section 11.3
- One Cell in portrait mode: section 11.4
- Three Cells in portrait mode ('Meeting Room'): section 11.5

Note: *In all examples, the Cell sizes are 1920 x 1080, with a 10 pixel bezel between adjacent Cells.*

11.1 Single Cell, landscape mode

This is the simplest Cell display topology:

Single Cell, landscape mode

11.1.1 Config.txt

By default, Cornerstone assumes that Cells are installed in landscape mode, so you do not need to rotate the tracking data. Also, with only one Cell outputting tracking data, you do not need to combine the tracking data.

For example, if the Cell IP address is 10.0.0.1, add the following block to config.txt.

```
NetBridge {
  host = "10.0.0.1"
  input-translate = "0 0"
}
```

11.1.2 Screen.xml

This display topology only uses a single graphics card and does not need bezel compensation. You therefore only need single `window` and `area` elements in screen.xml.

```
<!DOCTYPE mtdoc>
<MultiHead type="">
   <window type="window">
      <location type="">0 0</location>
      <size type="">1920 1080</size>
      <area type="">
         <location type=" ">0 0</location>
         <size type=" ">1920 1080</size>
         <graphicslocation type="">0 0</graphicslocation>
         <graphicssize type="">1920 1080</graphicssize>
      </area>
   </window>
</MultiHead>
```

11.2 Two Cells, landscape mode

This Cell display topology looks like this:

Two Cells, landscape mode

11.2.1 Config.txt

By default, Cornerstone assumes that Cells are installed in landscape mode, so you do not need to rotate the tracking data. But each Cell outputs tracking data in its own coordinates, so you must combine the tracking data into single coordinate system. For example, if the Cell IP addresses are 10.0.0.1 and 10.0.0.2, add these blocks to config.txt.

```
NetBridge {
  host = "10.0.0.1"
  input-translate = "0 0"
}
NetBridge {
  host = "10.0.0.2"
  input-translate = "1930 0"
}
```

Note that you need to translate tracking data received from the right-hand Cell, shifting its x coordinate to the right by 1930 pixels. For an explanation of how to combine multiple streams of tracking data, see section 10.5.2.

11.2.2 Screen.xml

This display topology only uses a single graphics card. There is a 10 pixel vertical bezel between the two Cells, but you use the proprietary tool to configure the operating system and add bezel compensation.

In screen.xml, you need to define a single window, physical screen area and logical screen area, all extending across both Cells. Pay attention to the *size* of each element:

- Set the window *size* to 3850 x 1080 pixels, where the width is two Cell widths plus the bezel ie 3850 = 1920 + 10 + 1920 pixels.
- Also set the *size* of the physical screen area to 3850 x 1080 pixels.
- Also set the *size* of the logical screen area to 3850 x 1080 pixels.

For each element, its *location* is the standard 0,0.

```
<!DOCTYPE mtdoc>
<MultiHead type="">
   <window type="window">
      <location type="">0 0</location>
      <size type="">3850 1080</size>
      <area type="">
         <location type=" ">0 0</location>
         <size type=" ">3850 1080</size>
         <graphicslocation type="">0 0</graphicslocation>
         <graphicssize type="">3850 1080</graphicssize>
      </area>
   </window>
</MultiHead>
```

11.3 Six Cells, landscape mode ('Board Room')

A Board Room display comprises two rows of three 1920 x 1080 Cells, driven by two quad-head GPUs. One GPU drives top row Cells 1, 2, and 3; a second GPU drives bottom row Cells 4, 5, and 6. There are two vertical bezels and one horizontal bezel.

Cell 1 (GPU 1)	**Cell 2** (GPU 1)	**Cell 3** (GPU 1)
Cell 4 (GPU 2)	**Cell 5** (GPU 2)	**Cell 6** (GPU 2)

Board Room display: six Cells, landscape mode.

11.3.1 Config.txt

By default, Cornerstone assumes that Cells are installed in landscape mode, so you do not need to rotate the tracking data. But each Cell outputs tracking data in its own coordinates, so you must combine the tracking data into single coordinate system. For example, if the Cell IP addresses are 10.0.0.1 through 10.0.0.6, add these blocks to config.txt.

```
NetBridge {
  host = "10.0.0.1"
  input-translate = "0 0"            /* Cell 1   */
}
NetBridge {
  host = "10.0.0.2"
  input-translate = "1930 0"         /* Cell 2   */
}
NetBridge {
  host = "10.0.0.3"
  input-translate = "3860 0"         /* Cell 3   */
}
NetBridge {
  host = "10.0.0.4"
  input-translate = "0 1090"         /* Cell 4   */
}
NetBridge {
  host = "10.0.0.5"
  input-translate = "1930 1090"      /* Cell 5   */
}
NetBridge {
  host = "10.0.0.6"
  input-translate = "3860 1090"      /* Cell 6   */
}
```

Note that you need to translate the tracking data from each Cell by shifting its origin right and down:

- **X coordinates:** For Cells 2 and 5 in the second column, shift their x coordinates to the right by 1930 pixels (one Cell width plus one vertical bezel). For Cells 3 and 6 in the third column, shift their x coordinates to the right by 3860 pixels (two Cell widths plus two vertical bezels).
- **Y coordinates:** For Cells 4, 5 and 6 in the bottom row, shift their y coordinates downwards by 1090 pixels (one Cell height plus one horizontal bezel).

For an explanation of how to combine multiple streams of tracking data, see section 10.5.

11.3.2 Screen.xml

To compensate for the vertical bezels, you can configure the GPU driver (see section 10.3), but to compensate for the horizontal bezel between the top and bottom rows you must edit screen.xml.

You must define a separate window and physical screen area for each GPU. In this example, GPU 1 drives the three Cells in the top row while GPU 2 drives the three Cells in the bottom row. Add the following elements and values to screen.xml.

```
<!DOCTYPE mtdoc>
<MultiHead type="">

  <GPU_1 type="window">                 /* Top row window                       */
    <location type="">0 0</location>    /* Top row window location              */
    <size type="">5780 1080</size>      /* Top row window size                  */
    <Area1 type="area">                 /* Top row area                         */
      <location type="">0 0</location>  /* Top row physical screen location     */
      <size type="">5780 1080</size>    /* Top row physical screen size         */
      <graphicslocation type="">0 0</graphicslocation>
                                        /* Top row logical screen location      */
      <graphicssize type="">5780 1080</graphicssize>
                                        /* Top row logical screen size          */
    </Area1>
  </GPU_1>

  <GPU_2 type="window">                 /* Bottom row window                    */
    <location type="">0 1080</location> /* Bottom row window location           */
    <size type="">5780 1080</size>      /* Bottom row window size               */
    <Area1 type="area">                 /* Bottom row area                      */
      <location type="">0 1080</location>
                                        /* Bottom row physical screen location */
      <size type="">5780 1080</size>    /* Bottom row physical screen size      */
      <graphicslocation type="">0 1090</graphicslocation>
                                        /* Bottom row logical screen location  */
                                        /* offset vertically by +10 from       */
                                        /* physical location ie, 1080+10=1090  */
      <graphicssize type="">5780 1080</graphicssize>
                                        /* Bottom row logical screen size      */
    </Area1>
  </GPU_2>

</MultiHead>
```

Screen.xml with horizontal bezel compensation added by the `<graphicslocation>` element in the GPU_2 window.

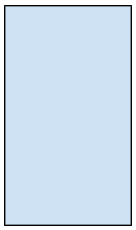
Here, the GPU_1 window and physical screen `area` extend across the top row of Cells, while the GPU_2 window and physical screen `area` extend across the bottom row.

The logical screen area for the top row, defined with `graphicslocation` and `graphicssize`, exactly matches the location and size of the physical screen area. But the logical screen area for the bottom row is located 10 pixels *below* the corresponding physical screen area. This means a 6mm horizontal strip of application graphics is now hidden by the bezel, but the overall visual effect will look more natural.

For a full explanation of the elements and values in this topology, see section 10.2.4.

11.4 Single Cell, portrait mode

In this simple display topology, the Cell is rotated clockwise 90° to portrait mode.



Single Cell, portrait mode

11.4.1 Config.txt

When Cells are rotated, you need to rotate and translate the tracking data coordinates. To make the tracking coordinate origin (0,0) match the upper left corner of the rotated screen, add the following blocks to config.txt:

```
NetBridge {
  host = "10.0.0.1"
  input-rotate = "90"         /* tracking data rotated by +90 degrees */
  input-translate = "1080 0" /* x coordinate translated to right     */
                              /* by +1080                             */
}
```

Rotation around the Cell upper left corner is applied first, while translation to the right is applied second. However, the order of `input-rotate` and `input-translate` in config.txt does not matter.

For an explanation of how to rotate streams of tracking data, see section 10.5.3.

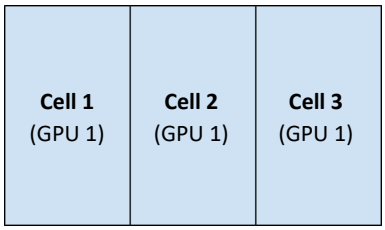
11.4.2 Screen.xml

This display topology only uses a single graphics card and does not need bezel compensation. You therefore only need single `window` and `area` elements in screen.xml.

```
<!DOCTYPE mtdoc>
<MultiHead type="">
   <window type="window">
      <location type="">0 0</location>
      <size type="">1080 1920</size>
      <area type="">
         <location type=" ">0 0</location>
         <size type=" ">1080 1920</size>
         <graphicslocation type="">0 0</graphicslocation>
         <graphicssize type="">1080 1920</graphicssize>
      </area>
   </window>
</MultiHead>
```

11.5 Three Cells, portrait mode ('Meeting Room')

A Meeting Room display comprises three Cells rotated clockwise 90° to portrait mode. This topology results in two vertical bezels, both 10 pixels in width.



Meeting Room display: three Cells, portrait mode.

11.5.1 Config.txt

When Cells are rotated, you must rotate and translate the tracking data coordinates to make the tracking coordinate origin (0,0) match the upper left corner of each rotated screen. Also, each Cell outputs tracking data in its own coordinates, so you must combine the tracking data into single coordinate system.

For example, if the Cell IP addresses are 10.0.0.1 through 10.0.0.3, add the following three blocks to config.txt:

```
NetBridge {
  host = "10.0.0.1"
  input-rotate = "90"          /* tracking data rotated by +90 degrees   */
  input-translate = "1080 0"   /* x coordinate shifted right by +1080    */
}
NetBridge {
  host = "10.0.0.2"
  input-rotate = "90"          /* tracking data rotated by +90 degrees   */
  input-translate = "2170 0"   /* x coordinate shifted right by +2170    */
}
NetBridge {
  host = "10.0.0.3"
  input-rotate = "90"          /* tracking data rotated by +90 degrees   */
  input-translate = "3260 0"   /* x coordinate shifted right by +3260    */
}
```

Rotation around the Cell upper left corner is applied first, while translation to the right is applied second. However, the order of `input-rotate` and `input-translate` in config.txt does not matter.

When a Cell is rotated 90° clockwise, its tracking coordinate origin is shifted to the left. To compensate for this, you must shift the origin back to the right:

- For Cell 1, shift the x coordinate 1080 pixels to the right. This amount is the width of a single Cell in portrait mode.
- For Cell 2, shift the x coordinate 2170 pixels to the right. This amount is the width of two Cells plus a bezel ie, 1080 + 10 + 1080.
- For Cell 3, shift the x coordinate 3260 pixels to the right. This amount is the width of three Cells plus two bezels ie, 1080 + 10 + 1080 + 10 +1080.

For an explanation of how to rotate streams of tracking data, see section 10.5.3.

11.5.2 Screen.xml

To compensate for the vertical bezels, you can configure the GPU driver (see section 10.3), so you do not need to add bezel compensation in screen.xml.

Also, this display topology uses a single graphics card, so you only need to define a single window, physical screen area and logical screen area, all extending across the three Cells.

Add the following elements and values to screen.xml.

```
<!DOCTYPE mtdoc>
<MultiHead type="">

  <GPU_1 type="window">
    <location type="">0 0</location>
    <size type="">3260 1920</size>       /* window width = 3260               */
    <area type="area">
      <location type="">0 0</location>
      <size type="">3260 1920</size>     /* physical screen area width = 3260 */
      <graphicslocation type="">0 0</graphicslocation>
      <graphicssize type="">3260 1920</graphicssize>
                                         /* logical screen area width 3260    */
    </area>
  </GPU_1>

</MultiHead>
```

Pay attention to the *size* of each element:

- Set the window *size* to 3260 x 1920 pixels, where the width is three Cell widths plus two bezel widths ie 3260 = 1080 + 10 + 1080 + 10 + 1080 pixels.
- Also set the *size* of the physical screen area to 3260 x 1920 pixels.
- Also set the *size* of the logical screen area to 3260 x 1920 pixels.

For each element, its *location* is the standard 0,0.

12 Cell firmware

12.1 Updating the Cell firmware

MultiTouch releases updates for MultiTaction devices to improve tracking quality and other functions. Cells include a self-update feature in the OSD. To update the firmware:

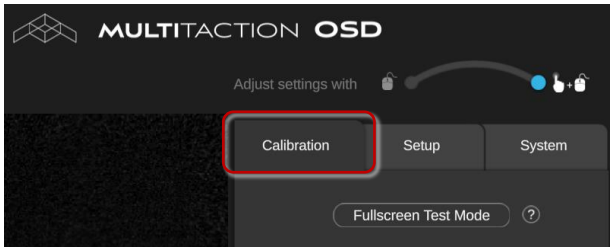
1. Go to the System tab and tap the Update button.
2. Tap the 'Check for updates' button.

   This causes the software to connect to the MultiTaction update server
3. Start the update and install the latest firmware version.

   If a network connection to the server cannot be made, an error is shown on the screen.
4. The update procedure typically takes 20 minutes, but the time may vary depending on the speed of your network connection. During the update procedure, the Cell may temporarily flicker and reboot, depending on the contents of the update.
5. *(Applies only if upgrading from firmware version 1.9 to 2.0)* Set the tracking parameters to their default values.

   Note: *Default values for some tracking parameters have been optimized in firmware 2.0 for more accurate tracking results. However, existing values for tracking parameters remain unchanged when you update the Cell firmware. We therefore recommend that you manually reset all parameters to the new default values when updating to firmware 2.0. This ensures the new default values are used instead of any 'old' default values from firmware 1.9 or earlier.*

   a. Display the OSD and tap the Calibration tab.

   OSD Calibration tab.

   b. Go to the pane for the tracking feature you want: Fingers, Markers or Pens.

   c. Turn on the tracker by moving the On-Off slider to On. For example:

   OSD Calibration tab, Markers pane, On-Off slider.

   d. Tap the Defaults button to set the tracking parameters to their default values.

   OSD Calibration tab; Fingers pane with Defaults button.

   Warning: *Do not turn off the Cell while updating the firmware.*

13 Remote management of Cells

MultiTaction hardware has several methods for remote management of Cells. These include the Site Manager tool (section 13.1) and Remote Control commands that enable integration with third party remote management systems (section 13.2).

13.1 Site Manager

MultiTaction Site Manager is an application for controlling a collection MultiTaction Cell displays. It can connect to any number of MultiTaction Cells and application computers to enable remote setup and control of multi-touch installations. Its role is to assist systems integrators, IT managers and content providers during the whole deployment life-cycle.

For more information, visit www.multitaction.com or contact sales@multitaction.com.

13.2 Remote Control commands

MultiTaction Cells support Remote Control commands that can be used to control Cells through third party remote management systems, such as systems provided by AMX or Crestron.

Notes

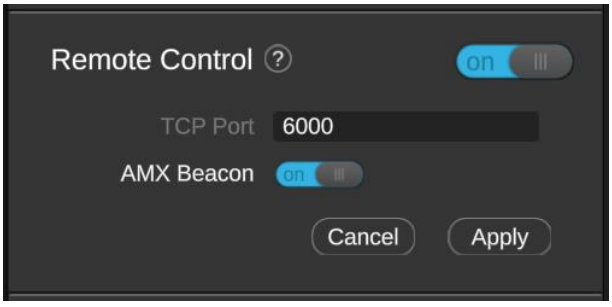
- *AMX manufactures hardware and software for video switching and control devices. For details, see www.amx.com.*
- *Crestron manufactures advanced control and automation technology. For details, see* www.crestron.com.

13.2.1 Command server and AMX Beacon

A remote management system comprises a command server (for receiving and handling the remote control commands) and AMX Beacon (used for device discovery).

The command server and AMX Beacon are disabled by default on MultiTaction Cells. To enable them, follow these steps:

1. Display the OSD and tap the Setup tab.
2. Go to the Remote Control section pane.



OSD Setup tab, Remote Control pane

3. Set the main Remote Control slider to On.

   The command server is now enabled and reachable on TCP port 6000.

4. Set the AMX Beacon slider to On.
5. Click Apply.
6. Tap the Save button to save the Remote Control settings.

   You may need to scroll to the bottom of the Setup tab to see the Save button.

13.2.2 Version details

- **Compatibility with AMX protocol:** MultiTaction Cells currently support Remote Control commands compatible with version 1.1 of the AMX protocol (released February 2013).
- **MultiTaction firmware requirement:** To support for AMX 1.1-compatible Remote Control commands, MultiTaction Cells must be running firmware version 1.9.1-taction17 or later.

13.2.3 Command string requirements

Note the string requirements for Remote Control commands:

- Strings must consist of only **ASCII** characters.
- The maximum length for send and return strings must be less than **256** characters.
- Strings are terminated with **CR**, **LF**.

13.2.4 Command string requirements

Remote Control commands use four string types:

- **Command strings**: client → MultiTaction Cell

  ```
  <Command name>[param 0][param 1...]<Line feed>
  ```

  Command name and parameters are separated by an ASCII space character. Example: `STANDBY 1\r\n` is a `STANDBY` command with one parameter `1`.

  Query commands are prefixed with '`?`'. Example: `?STANDBY\r\n` queries the standby status.

  All integer parameters are encoded as ASCII decimal character arrays.

- **ACK strings:** MultiTaction Cell → client

  ```
  OK<Line feed>
  ```

  Server will send ACK response when it has parsed a command from the client and there have been no errors when processing the command.

- **Query response strings:** MultiTaction Cell → client

  ```
  <Request>=<Response, no line feeds><Line feed>
  ```

  Server will send response to query commands with one response string per query. On normal commands, only ACK is returned, unless there was an error. Request is the original command, excluding the beginning question mark '`?`'.

- **Error strings:** MultiTaction Cell → client

  ```
  Error <Human readable description><Line feed>
  ```

  In the case of an internal error or invalid / unknown command, the server will send an error string to the client.

13.2.5 Command reference

MultiTaction Cells support the following Remote Control commands and responses:

Command	Example response	Description
?VERSION	PROTOCOL: 1.1, VM1: B2.7, DCU: 7 7 7 7 7 7 7, DGU: 10, FIRMWARE: 1.9.1-taction17	Query the AMX control protocol, hardware and firmware version information.
?VERSION <COMPONENT>	VERSION VM1=B2.7 (for query ?VERSION VM1)	Version information of the given component. Return response only includes the version string without any prefix. List of components: PROTOCOL, FIRMWARE, VM1, DGU, DCU
?MODEL	MODEL=MT550L	Query the hardware model.
?STANDBY	STANDBY=1|0	Is the screen on standby?
STANDBY 1		Enter standby mode (turn off LCD, IR, Cameras, Backlight)
STANDBY 0		Recover from standby mode
?DPMS	DPMS=ON | STANDBY | SUSPEND | OFF	Query the DPMS state of the panel / internal PC graphics card.
LCD-POWER 0		Turn off LCD power.
LCD-POWER 1		Turn on LCD power.
?BRIGHTNESS	BRIGHTNESS=0-255	Query LCD backlight brightness.
BRIGHTNESS <X>		Set LCD backlight brightness to X. X is integer between 0 and 255.
?LOGO-TIMEOUT	LOGO-TIMEOUT=<seconds>	Query the VM1 logo timeout in seconds (how long the logo is displayed before the screen is blanked)
LOGO-TIMEOUT <X>		Set the VM1 logo timeout to X seconds.
?VIDEO-INPUT-AUTO	VIDEO-INPUT-AUTO=1 | 0	Query the status of VM1 video input auto-selection.
VIDEO-INPUT-AUTO 1		Turns on video input auto-selection.
VIDEO-INPUT-AUTO 0		Turns the auto-selection off.
?VIDEO-INPUT-PRIORITY	VIDEO-INPUT-PRIORITY=EXTERNAL_DVI | INTERNAL	Query the priority of VM1 video input auto-selection.
VIDEO-INPUT-PRIORITY EXTERNAL_DVI		Give priority to external video input.
VIDEO-INPUT-PRIORITY INTERNAL		Give priority to internal PC (OSD).
?VIDEO-INPUT	VIDEO-INPUT=EXTERNAL_DVI | INTERNAL | TEST_IMAGE | LOGO	Query active video input.
VIDEO-INPUT <INPUT>		Choose video input, where <INPUT> is one of ?VIDEO-INPUT responses. Overrides and turns off auto-selection.

Command	Example response	Description
?TEMPERATURE VM1	TEMPERATURE VM1=<Integer, °C>	Query the VM1 temperature °C.
?TEMPERATURE GPU	TEMPERATURE GPU=<Integer, °C>	Query the GPU temperature °C.
?IMAGE-DATA-INFO RX	IMAGE-DATA-INFO RX=<Integer, bytes>	Query the internal image stream statistics. Returns the number of bytes the internal PC has received in the camera stream data from DGU since last reboot.
?IMAGE-DATA-INFO TX	IMAGE-DATA-INFO TX=<Integer, bytes>	Number of bytes sent to DGU since last reboot.
?IMAGE-DATA-INFO ERRORS	IMAGE-DATA-INFO ERRORS=<Integer, number of errors>	Number of detected errors in the camera data stream since last reboot.
?NETWORK-INFO RX	NETWORK-INFO RX=<Integer, bytes>	Number of bytes received from the external network adapter since last reboot.
?NETWORK-INFO TX	NETWORK-INFO TX=<Integer, bytes>	Number of bytes sent to external network adapter since last reboot.
?NETWORK-INFO ERRORS	NETWORK-INFO ERRORS=<Integer, number of errors>	Number of detected errors in the external network adapter since last reboot.
REBOOT		Restarts the embedded computer, does not cycle power.

14 Frequently asked questions

14.1 Markers

14.1.1 Marker size

What are the size limitations of markers?

Marker sizes are described in section 8.5.

14.1.2 Markers on a phone screen

MultiTouch Cells can track markers that are printed on paper. But can they track markers displayed on a mobile phone's screen?

We have tested this briefly. Generally, the picture on the mobile phone screen does not show up in the infrared tracking. Therefore, this approach is not currently possible.

14.1.3 Markers and finger tracking

When I keep one or more markers on screen, the finger tracking gets poor sometimes. Why is that?

When a marker is put on the screen, Cornerstone will stop the automatic background calibration. This is done to avoid the situation where the marker gets burnt into the calibration image. As a consequence of this, the background IR image will not change at all, which may cause poor finger tracking in extended use.

To counter this issue, try to keep tracking conditions stable when using markers. Alternatively, lift all markers from the screen occasionally, so that the background calibration can adapt to the lighting conditions.

14.2 Cornerstone

14.2.1 C++ and application performance

Compared to easy scripting environments like Flash/Air/AS3, VVVV, Python, Ruby etc., how fast is the C++ really?

Usually, applications written with C++ and OpenGL are 10-100 times faster than applications written with Flash. As a simple benchmark, we have seen Flash- and VVVV-based applications stutter even when there are only a few items on the display.

In practice, this means that applications written with scripting languages are often optimized heavily after the application is feature-complete. This optimization typically requires great knowledge of the particular scripting environment, so the ‘easy’ Flash/Air/VVVV environments often turn out to require high levels of technical skill unless the application is extremely simple. This is a hidden cost that often surprises both the application developer and the end customer. By using a high-performance engine, this problem is removed.

14.2.2 Windows Touch API with multiple Cells

We would like to use the Windows Touch API with multiple Cells, but the Cornerstone driver only supports a single Cell. Why is there such a limitation?

Due to a Microsoft decision, Windows Touch input is limited to the primary monitor. According to Microsoft, Windows 7 native touch does not support multiple touch-screens simultaneously. Officially, therefore, if you want to use multiple Cells (or other displays) together, you cannot use the Windows Touch API because Windows does not support native touch on more than one screen.

However, you may be able to overcome this limitation by configuring the GPU driver to provide a single virtual display that incorporates multiple screens. For details, contact MultiTaction Support: www.multitaction.com/support-services

14.2.3 Working with Adobe Flash

How do I develop multi-touch applications with Adobe Flash?

We do not support Flash directly. If needed, you can get touch information to Flash applications by using the XML stream or TUIO.

14.3 Cell internal computer

14.3.1 MultiTaction hard drive corruption

There is a computer inside the MultiTaction Cell. Will the hard disk of the computer get corrupted if I turn off the power with telling the computer to shut down?

For *stackable* MultiTaction Cells, the hard disk is mounted in a read-only mode. This means that the device can be generally rebooted or shut down, without any worry of the file system being corrupted. As an exception, the file system is turned into read-write mode for saving parameters, and for software updates. During these operations the display must not be shut down.

14.3.2 MultiTaction internal computer in stackable models

How do I access the internal computer of stackable MultiTaction Cells?

The internal computer cannot be used freely by customers. If you still feel that you'd like to use the internal computer, please contact MultiTouch Sales.

14.4 Reference

14.4.1 Best operating system

What is the best OS for the application computer?

The “best" depends on what you are doing, and what kind of limitations you have regarding the compatibility with various libraries.

Please see the operating system comparison in section B.6.

14.4.2 Image formats

Which image formats can I use with Cornerstone applications?

Cornerstone supports the following image formats:

- PNG
- JPEG
- DDS files in DXT 1/3/5 formats

What are the image requirements?

- All images, icons, and backgrounds must be in PNG format.
- All photographs must be in JPEG or PNG format.
- Image resolution must be high enough for use when graphics are scaled up. Typically, this means at least 1920 x 1080 for images that are zoomable.

14.4.3 Video formats

Which video formats can I use with Cornerstone applications?

The list of supported video formats and codecs is extensive. On Linux computers, run the following commands to see the full lists:

```
/opt/multitouch-libav1/bin/avconv -formats
/opt/multitouch-libav1/bin/avconv -codecs
```

What are the recommended video formats?

- AVI with H.264 encoding.
- All video sound tracks must be 44.1kHz.

Which is the most reliable video format to use?

The most reliable video format is generally an AVI file with motion JPEG (mjpeg) video compression and uncompressed linear PCM audio at 44.1 kHz (ie, CD-quality audio). Motion JPEG is not the most effective codec, so normally we use DivX4 (ie, MP4) in an AVI container. For the best compression quality, H.264 is often used.

What are requirements for video seeking?

Video seeking requires I-frames in the video file. For accurate seeking, the file must contain at least one I-frame and audio chunk per second. If the I-frames (or audio chunks) are missing, the playback engine is less accurate when finding the seek location.

14.4.4 Graphics cards for application PC

Which graphics cards do you recommend?

Cornerstone supports NVIDIA graphics cards. See section B.7 for example setups for the application computer.

For appropriate performance, we do not recommend low-end graphics cards.

14.4.5 Other programming languages

I am interested in using CornerStone with Java, Python, Ruby, Processing, Lua, .Net, WPF, Delphi, or some other programming environment. Can I do this?

Yes, you can:

- **TUIO or XML streams**

  TUIO or XML streams can be used to import the MultiTaction finger and hand information to practically any programming environment, just by receiving data from a socket. The choice of protocol depends on the feature set you need (XML providing more comprehensive feature set) and compatibility requirements (many projects already use TUIO).

- **Direct bindings**

  Direct bindings to the Cornerstone APIs can be created as well, providing a more direct access to the underlying APIs, with possibility to integrate into the existing Cornerstone frameworks, if necessary.

  Bindings can be generated with **Swig**, which is a Simple Wrapper Interface Generator. Swig can produce bindings for several languages, from AllegroCL to Octave; see www.swig.org/

14.5 Other

14.5.1 Images are torn in multiple-Cell setups

We are using multiple graphics cards to drive a large number of displays. When we are moving objects, they are sometimes torn in the middle. Why is this?

The explanation is very technical. When multiple graphics cards are in use, they typically do not share the same pixel clock, resulting in different timings of the picture scan-out process. This means that some GPUs might update the image (ie, switch the front and back buffers) while the frame is being sent out to the display. This is a limitation of the graphics cards.

The solution is to use more expensive, professional graphics cards which have a 'GenLock' feature that overcomes this problem. Relevant cards can be found in the NVIDIA Quadro series. These cards typically cost more than comparable consumer cards and their configuration is more complex. For more information on the topic, contact NVIDIA.

14.5.2 Microsoft Surface SDK

Can I use Microsoft Surface SDK with MultiTouch Cells? Is there anything special I would need to consider?

Most applications that were written with the Surface SDK will work instantly with MultiTouch Cells, using the Windows native touch driver. This driver implements the multi-touch interface found in Windows 7 and later, so that finger-tip information is transferred from the Cells to the Windows operating system.

Advanced features like markers and hand information are not passed via this interface. To pass marker information, use either TUIO or the XML stream.

14.5.3 Cell as a plain display

Is it possible to connect the Cell to an external computer and use it as a monitor?

Yes. You can do this if, for example, you only want to use the Cell as an extra full-HD computer display. Note that Cells only take DVI input.

14.5.4 Longer cables

I would like to use really long cables between the computer and the display, around 50 meters (150 feet). Is this possible?

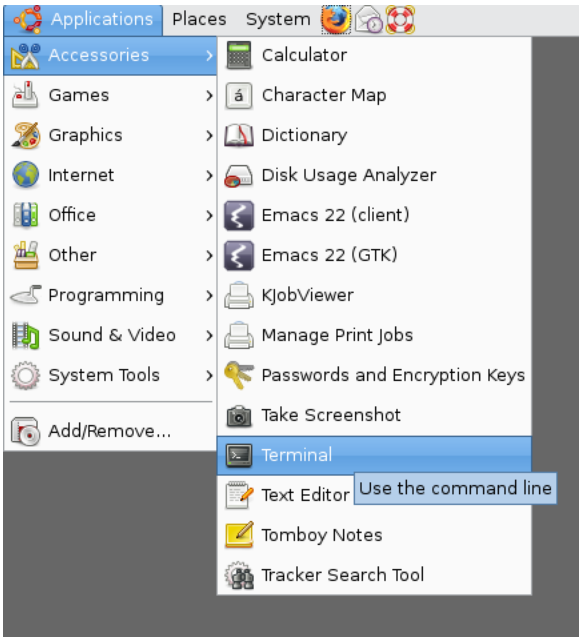
To do this, you would need to use a DVI extender to extend the DVI cable between your computer and the display.

In practice, DVI extenders often cause significant problems. For example, a DVI extender may lose the display EDID information, which is needed by the operating system to drive the display with correct signals.

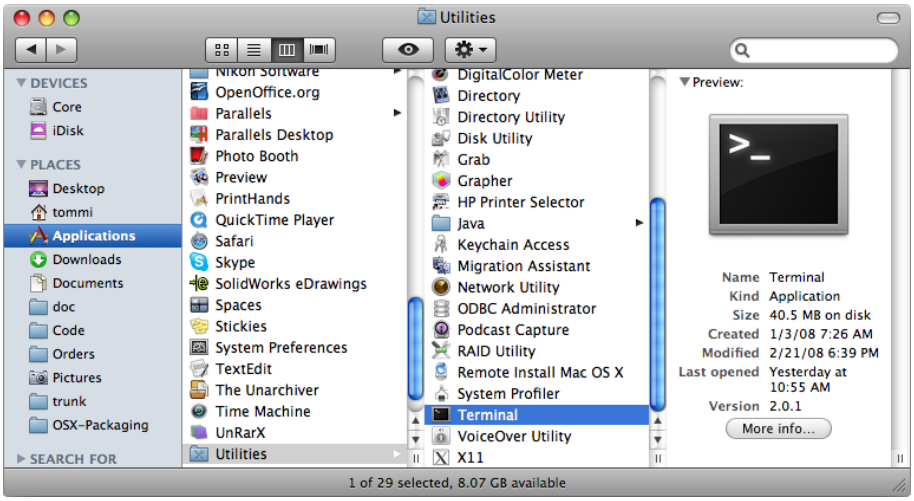
14.5.5 Starting terminal

Many operations seem to require a "terminal". How do I start this?

- On **Linux**, start terminal from the system menus:



- On **OS X**, use Finder to find the terminal application in the Applications/Utilities folder:



15 Troubleshooting

15.1 General

15.1.1 Display lag and ghosting

When I drag images across the screen, there is a disturbing lag and strong trailing. How do I fix this?

There can be different causes for the issue. Below are some known causes for this problem.

- **Weak electricity network.** Especially with multiple Cells, a MultiTaction installation takes plenty of electricity. We have seen cases where poor system performance is remedied by simply arranging more power to the Cells and the computer. Note that a weak or fluctuating power grid does not stop the Cell/computer operation completely, but just makes it more sluggish and random.
- **Cold.** Using the Cells in cold environments (below 10C/50F) can cause problems with the LCD panels, since the response time of the LCD panel increases as the temperature drops. In general, touch-displays are not very useful when temperature goes below 10C/50F because the surface is then uncomfortable to the touch, but sometimes trade show construction is done in such temperatures.

15.1.2 Sound card not working

I have a sound card that is not working correctly. There is no sound. What to do?

First, confirm that the sound card is detected by the PortAudio device handlers. To do this, use the `ListPortAudioDevices` application found in the bin folder.

Also, confirm that you are using the correct sound card API. To do this, run any video-enabled application in verbose mode and check the application debug output.

15.1.3 Cell is not responding to touch

A Cell is not responding when I touch the screen. This happens whether I use my fingers, marker, or an infrared pen. How do I fix this?

Verify the Cell is not using a prohibited IP address. The prohibited subnet is 192.0.2.x.

Although the OSD prevents you from manually assigning an IP address in the forbidden range, your DHCP server may do so. If your DHCP server has assigned an IP in the forbidden range to the Cell, you will need to manually assign a legitimate IP address.

To see which IP address has been applied to your Cell and, if necessary, manually assign an IP address, check the OSD Network settings; see section 8.3.1.

15.2 Windows

15.2.1 High latency

There is a lot of latency between hand movement and its effect on the screen. I am using Windows 7, 64-bits. How do I fix this?

This is a rare bug and the exact cause is unknown. Running **DPC Latency Checker 2.0** seems to solve this problem, at least for the known cases:

http://www.thesycon.de/deu/latency_check.shtml

Based on information on the DPC web site, the latency problems are probably caused by faulty device drivers. Any misbehaving device driver (display, USB, network, Firewire and so on) may introduce extra latency for the whole system.

15.3 Linux

15.3.1 Graphics performance

The graphics on my system execute very slowly, unlike in the demos. How do I fix this?

The most usual cause is the lack of proper graphics hardware or drivers. For hardware, you should have a reasonably modern graphics card from NVIDIA.

15.3.2 Multiple monitors

I am using a Cell along with another (main) monitor. The graphics window does not open on the Cell, but on the main display. How do I get the graphics on the Cell?

The problem stems from the Gnome display manager that limits the locations of application windows.

The usual solution is to enable the full screen mode, and place the OpenGL viewport to the desired location:

1. Open the file ~/.MultiTouch/screen.xml
2. Set the `fullscreen` field to 1
3. Edit the *window* → *area* → *location* to match the Cell graphics location.

15.3.3 Videos not playing

The videos do not play. They get stuck in the first frame. How can I fix this?

The most usual cause is that the audio output is not working properly. (Video playback is synchronized to the audio output.) Most probably, some other application is already using the audio device you intend to use. Often this is the system sound daemon, which can be disabled.

16 Maintenance and cleaning

16.1 Prevent screen burn-in

Caution

- *Static images drawn on-screen for a long period of time can create afterimages that are "burnt" on the display and remain permanently.*
- *Run an application that regularly changes the screen's contents, alternating from static images to moving pictures (this helps refresh the display's liquid crystal). Periodical changes in background color and character are also recommended.*
- *Enable a screen-saver (with moving pictures or a black pattern).*
- *Avoid backgrounds and content with markedly contrasting luminance.*

Avertissement

- *Les images statiques restant sur l'écran pendant une longue durée, peuvent engendrer des images rémanentes « calcinées » sur celui-ci, y restant en permanence.*
- *Exécuter une application modifiant régulièrement le contenu de l'écran, alternant entre images statiques et images animées (ce qui aidera à rafraîchir le cristal liquide de l'écran). Des changements périodiques de la couleur et du caractère du fond, sont également recommandés.*
- *Utiliser un économiseur d'écran (avec des images animées ou un motif noir).*
- *Éviter en principe les fonds et contenus dont la luminosité contraste nettement.*

16.2 Cell maintenance or repair

Technical maintenance or repair of the Cell must always be conducted by personnel approved by Multitouch.

Danger

- *Opening the device enclosure causes immediate risk of electrocution, severe personal injury and property damage.*
- *Never open the device enclosure.*

Danger

- *L'ouverture du boîtier de l'appareil entraîne un risque immédiat d'électrocution, de blessures graves et de dommages matériels.*
- *Ne jamais ouvrir le boîtier de l'appareil.*

16.3 Cleaning the Cell

1. Spray the glass with a standard window cleaning solution, such as Windex® Multi-Surface Cleaner (colorless)
2. Immediately remove the cleaning solution using a lint free cloth

Warning

⚠ *Do not use ammonia-based cleaning solutions*

⚠ *Do not use abrasive cleaners*

⚠ *Do not use abrasive applicators*

⚠ *Do not drench the display surface or edges with the cleaning liquid*

⚠ *Do not use scrapers, knives or other sharp or hard objects for cleaning the display glass*

Avertissement

⚠ *Ne pas utiliser des solutions de nettoyage à base d'ammoniaque*

⚠ *Ne pas utiliser de produits nettoyants abrasifs*

⚠ *Ne pas utiliser d'applicateurs abrasifs*

⚠ *Ne pas mettre de liquide de nettoyage sur la surface d'affichage ou les bords*

⚠ *Ne pas utiliser de grattoirs, couteaux ou autres objets pointus ou durs, pour nettoyer le verre de l'écran*

16.4 Warning to service personnel

- Always connect Cells to the mains power supply using a 3-wire cord and power strip with a protective earth. Failure to do so may cause an electric shock when using the Cell!
- When connecting other equipment to the Cell, make sure all equipment is disconnected from the mains. Failure to do so may damage the Cell through ground leakage currents from and to other devices.

Caution to service personnel

⚠ *Double pole / neutral fusing.*

⚠ *Maintain with care.*

Avertissement destine au personnel de service

⚠ *Fusible au neutre / bipolaire.*

⚠ *Entretenir avec soin.*

16.5 Replaceable battery

Please contact MultiTaction Support for instructions on how to replace the lithium battery: www.multitaction.com/support-services

The battery is positioned in the service access area on the PC mainboard. The battery type is CR2032 (Fujitsu D3243-S mainboard).

Caution

Risk of explosion if the battery is replaced with an incorrect type. Battery should be recycled when possible. Disposal of used batteries must be in accordance with local environmental regulations.

Avertissement

Risque d'explosion si la batterie est remplacée par un type incorrect. La batterie doit être si possible recyclée. L'élimination des piles usagées doit se faire en conformité à la réglementation environnementale locale.

Appendix A. Technical specifications

A.1 Display

	MT553UTB	MT555UTB
Display technology	TFT LCD ultra-thin bezel	TFT LCD ultra-thin bezel
Display size	55" 16:9 wide	55" 16:9 wide
Native Resolution	Full HD 1920 x 1080	Full HD 1920 x 1080
Display Colors	16.7 million	16.7 million
Display Active Area	1209 x 680 mm	1209 x 680 mm
Pixel Pitch	0.63 x 0.63 mm	0.63 x 0.63 mm
Pixel Response Time	8ms grey to grey	8ms grey to grey
Brightness	300 cd/m2	500 cd/m2
Contrast Ratio	4000:1 typical	4000:1 typical
Picture Frame Rate	60 Hz	60 Hz
Backlight	White LED matrix array	White LED matrix array
Viewing angle	178° (horizontal and vertical)	178° (horizontal and vertical)

A.2 Touch sensing hardware

	MT553UTB and MT555UTB
Touch technology	Computer Vision Through Screen
Tracking speed	Up to 200 fps
IR source and camera	Integrated Backlight Emitter Camera modules
IR wavelength	850 nm

A.3 Interfaces

	MT553UTB and MT555UTB
Video input	DVI-D; DisplayPort (disabled); x2 RCA (disabled)
Control input	Ethernet; USB Type-B (disabled)
Keyboard and mouse input	x3 USB 2.0 Type-A
Power connector	IEC C14

A.4 Electrical specifications

	MT553UTB, MT555UTB
Power supply	100-240 VAC 50/60 Hz
Mains fuse	Ø5 x 20mm 6.3A 250VAC T (Time-Lag)
Audio	None

A.5 Power consumption

	MT553UTB	MT555UTB
Typical	440W	470W
Maximum	510W	530W

A.6 Touch details

	MT553UTB and MT555UTB
Tracking Software	Embedded MultiTaction Extensible Hybrid Tracking Engine
Interaction methods	Single finger, multiple fingers, single hand, multiple hands and users, objects with optical markers, basic shapes, infrared pen. Click, hold and drag.
Hand recognition	Each finger orientation recognized and fingers identified to certain hand.
Object recognition (infrared pen)	Supported with 2D fiducial markers. Some basic shapes. Unlimited simultaneous pen and touch input.
Number of simultaneous touch Inputs	Unlimited
Number of simultaneous users	Unlimited (separate hands)
Positional accuracy	2mm, directional compensation
Relative accuracy	Sub-pixel accuracy
Touch latency	10 ms typical. Measured from touch to processed tracking data output. Overall latency is greater and depends on the computer, operating system, and application used.
Tracking data outputs / APIs	Multi-Format Tracking Output: Low level C++, TUIO, XML stream, Windows 7 Touch
Operating system support	Windows 7 or later; Linux- Ubuntu; Mac OS X
Development environment support	Most major programming languages and environments (MS Surface 2 SDK, TUIO-compatible systems). See MultiTouch Cornerstone documentation for further details.
Software Development Kit	Cornerstone SDK available for C++/OpenGL (Windows 7, Linux, Mac OS X)
Stackability	Any number of Cells can be stacked to run a single multi-touch application. However, the number of Cells for one computer is limited, based on the computer configuration.

A.7 Environment

	MT553UTB and MT555UTB
Operating temperature	0 to +35 °C
Storage temperature	-10 to +60 °C Allow at least two hours for temperature normalization before switching on the unit.
IP rating	20
Relative humidity	Non-condensing 80%

A.8 Dimensions and weight

	MT553UTB	MT555UTB
Size	1215 x 686 x 200 mm	1215 x 686 x 200 mm
Bezel width	1.9 mm (bottom & left) 3.8 mm (top & right)	1.9 mm (bottom & left) 3.8 mm (top & right)
Weight	41 kg (90 lbs) plus 5 kg for two legs	41 kg (90 lbs) plus 5 kg for two legs
Color	Black	Black

A.9 Display mounting

	MT553UTB and MT555UTB
VESA Mount	800 x 400 mm, M8 screws M8 screw thread length: Threads must be 14mm *plus* the thickness of the mount material.
Mounting positions	Vertical Horizontal Angled position Table position Display Array (2-24 units) Symmetric or asymmetric layouts

A.10 Cooling

	MT553UTB and MT555UTB
Cooling system	Forced air
Convection method	Low Noise Variable Speed Internal Fans
Noise level	40-45 dBa

A.11 Ventilation

Air passing through the Cell	MT553UTB and MT555UTB
Typical amount	220 m^3 / hour (at NTP conditions)
Maximum amount	250 m^3 / hour

A.12 Miscellaneous

	MT553UTB and MT555UTB
Agency approvals	CE
RoHS compliant	Yes
Expected lifetime	Over 60,000 h
Warranty	1 year limited (See warranty details, section 0)

Appendix B. Application computer: recommended setup

B.1 Sound cards

In most cases, the built-in audio cards of the host computer are good enough for playing the sounds required, and you do not need special audio hardware. The need for special hardware arises if:

- The built-in interface is not working correctly.
- The built-in interface has too weak audio quality.
- The built-in interface does not support enough audio channels.

In general, we recommend PCI/PCI-Express sound cards if the built-in audio interface is not sufficient for the project.

We discourage the use of USB sound cards, since they are often a cause of trouble. For example, an 8-channel audio interface might be able to generate only 6 channels in some circumstances. Due to problems like this, we recommend you do not use USB audio devices.

B.2 Graphics cards

Note the following:

- **OpenGL 4.1 or later:** The role of the graphics card depends on the application type. Because MultiTaction demo applications and calibration software use advanced OpenGL features, you must use graphics cards (or integrated graphics) from NVIDIA.

  These graphics cards and drivers must be compliant with OpenGL 4.1 or later. Note that the recommended graphics card, NVIDIA Quadro M5000, supports OpenGL 4.5.

- **Performance:** In some OpenGL-based applications with heavy graphics, the application performance may be dependent on the rendering speed of the card. However, we often run demos on a stock laptop, using a mid-range GPU.

- **Which graphics cards?** We currently recommend an NVIDIA Quadro M5000.

- **Multiple video outputs:** The NVIDIA Quadro M5000 has 4 video outputs per card, and a single card can support up to four Cells.

  For larger MultiTaction video wall solutions, we recommend two, three or four M5000 graphics cards, depending on the number of Cells. For example, the MultiTaction Lobby solution has 12 Cells and requires three M5000 graphics cards.

B.3 CPU

The impact of the CPU on system performance varies between different types of installations. The minimum CPU recommendation is a modern 3.5GHz processor (or greater) from Intel.

Cornerstone takes advantage of multiple CPU cores extremely well. For applications written with the MultiTouch Cornerstone OpenGL/C++, a quad core CPU is the best.

We currently recommend a single Intel Xeon E5-1650 CPU for video walls with up to three Cells. For larger video walls, such as the 12-Cell MultiTaction Lobby solution, we recommend dual Intel Xeon E5-2637 CPUs.

B.4 Memory

Memory tends to be a simple issue: either there is enough memory, or too little.

We recommend 16 GB RAM for standard multi-touch applications. This is enough to run all the standard demo applications that come with the Cornerstone software package. For larger video walls, such as the 12-Cell MultiTaction Lobby solution, we recommend 64 GB.

More memory may be needed to run applications with large in-memory data-sets. In these cases, the memory requirements depend entirely on how much memory the application consumes.

B.5 Hard disks

MultiTouch Cornerstone SDK takes less than 200MB of disk space. The third party support libraries and possible developer tools easily take 2-4GB of disk space, which is still very little compared to the volume of modern hard disks.

The size requirements of the hard disk depend entirely on the application needs. Unless there is a specific reason to get a large hard disk, you can simply buy a normal hard disk in the 240-1,000 GB range. We currently recommend a Samsung 480 GB SSD hard disk.

Some applications may have special requirements for the hard disk. For example, a video-intensive application might need a RAID stack or a fast SSD to be able to feed video data from the hard disk fast enough for the graphics. In these cases, there is no general rule; judge the performance requirements based on the applications.

B.6 Operating system

The choice of operating system is usually based on the skill set of the development team, client requirements and application requirements. Below, we have listed strong and weak points of OSX, Windows and Linux.

B.6.1 OS X

Pros:

- Easy software setup (no patches, no permission tricks)

Cons:

- Mac Pros are expensive
- Mediocre performance, compared to the hardware
- OpenGL drivers lag behind Windows and Linux

B.6.2 Windows

Pros:

- Good availability of third party software
- Often preferred by corporate network managers

Cons:

- Requires extra drivers and patching to get it to work
- Multiple-Cell installations typically require extra tweaking to get them to work

B.6.3 Linux

Pros:

- Good customization possibilities
- High performance

Cons:

- Hardware compatibility can be an issue
- May require special skills from the development/deployment team

B.7 Example setups

For illustration purposes, this section provides basic setups that can be used as a basis for ordering application computers.

B.7.1 Baseline application computer for 1 to 4 Cells (Windows or Linux)

We use this type of computer for most purposes. It has a good price/performance ratio and handles a range of uses. The recommended GPU supports up to 4 Cells (video heads).

- Processor: Intel Xeon E5-1650
- Memory: 16 GB RAM
- Hard disk: 480 GB SSD
- Graphics card: x1 NVIDIA Quadro M5000
- OS: Windows 10 or Linux Ubuntu 14.04 LTS

B.7.2 Application computer for 8 Cells (Windows or Linux)

To drive up to 8 Cells, we recommend a more powerful computer with two CPUs and two quad-head GPUs.

- Processor: x2 Intel Xeon E5-2637
- Memory: 64 GB RAM
- Hard disk: 480 GB SSD
- Graphics: x2 NVIDIA Quadro M5000 (each GPU has 4 video outputs)
- OS: Windows 10 or Linux Ubuntu 14.04 LTS
- Other: x8 active Mini DisplayPort to DVI adaptors

B.7.3 Application computer for 12 Cells (Windows or Linux)

To drive up to 12 Cells, we recommend a more powerful computer with two CPUs and three quad-head GPUs.

- Processor: x2 Intel Xeon E5-2637
- Memory: 64 GB RAM
- Hard disk: 480 GB SSD
- Graphics: x3 NVIDIA Quadro M5000 (each GPU has 4 video outputs)

  Note: *The M5000 is a dual-slot graphics card. The case and motherboard must have space for the three graphics cards.*
- OS: Windows 10 or Linux Ubuntu 14.04 LTS
- Other: x12 active Mini DisplayPort to DVI adaptors

See also issues affecting setups with multiple graphics cards; see section 14.5.1.

B.7.4 Application computer for 16 Cells (Windows or Linux)

To drive up to 16 Cells, we recommend a more powerful computer with two CPUs and four quad-head GPUs.

- Processor: x2 Intel Xeon E5-2637
- Memory: 64 GB RAM
- Hard disk: 480 GB SSD
- Graphics: x4 NVIDIA Quadro M5000 (each GPU has 4 video outputs)

  Note: *The M5000 is a dual-slot graphics card. The case and motherboard must have space for the four graphics cards.*
- OS: Windows 10 or Linux Ubuntu 14.04 LTS
- Other: x16 active Mini DisplayPort to DVI adaptors

See also issues affecting setups with multiple graphics cards; see section 14.5.1.

B.7.5 Laptop as application computer (OS X)

Apple laptops work well with MultiTouch Cornerstone. For this purpose, you can use a baseline MacBook Pro. For example:

- Processor: Intel Core i5 or i7
- Memory: 16GB RAM
- Hard disk: 200GB
- Graphics: NVIDIA
- Operating system: OS X Yosemite

B.7.6 Laptop as application computer (Windows or Linux)

There is a wide range of suitable laptops available. A typical setup could be:

- Processor: Intel Core i5 or i7
- Memory: At least 8GB RAM
- Hard disk: 200GB
- Graphics: Discrete GPU from NVIDIA
- Operating system: Windows 10 64-bit or Ubuntu Linux 64-bit

Appendix C. Config.txt and screen.xml syntax reference

C.1 Config.txt syntax

Cornerstone must be able to combine streams of tracking data from multiple Cells into a single coordinate system that matches the display topology. Also, Cornerstone assumes that Cells are in landscape mode. If Cells are installed in portrait mode, Cornerstone needs to transform the tracking data coordinates accordingly.

Config.txt provides Cornerstone with information about the number and orientation of Cells so it can correctly interpret touch coordinates in the tracking data that it receives, combining and transforming the tracking data as required.

The syntax for config.txt is as follows

`Globals` block

`binary-server-host`
: Defines the hostname for a network bridge server.

`binary-server-port`
: Defines the port number for the network bridge server.

`xml-handshake-port`
: Defines the port number for the XML handshake server socket. Flash applications need this.

`xml-server-host`
: Defines the hostname for the XML server socket.

`xml-server-port`
: Defines the port number for the XML server socket. The protocol is TCP/IP.

`NetBridge` block

`host`
: Contains the Cell IP address or `localhost`.

`port`
: Defines the port number of the NetBridge server running in the Cell.

`input-translate, "x y"`
: Defines the `"x y"` offset that gets added to the tracking data coordinates.

`input-rotate`
: Defines a clockwise rotation, in degrees, around the upper left corner of the tracking data coordinates.

  Note: *In Cornerstone 1.2.3 and earlier, the rotation is in radians.*

`input-scale , "xscale yscale"`
: Defines a scaling for the tracking data coordinates, with the upper left corner as the origin.

`TUIOSender` block. Defines parameters for TUIO tracking data.

`features`

Defines which tracking features that are enabled. To enable multiple features, add multiple keywords to the field. Options are:

`"fingers"` Sends fingertip locations

`"hands"` Enables the TUIO Hand Extension and send palm / finger connectivity information.

`"objects"` Sends marker information

`"verbose"` Prints `TUIOSender` output messages

The default value is `"fingers objects"`.

`address`

Defines the TUIO stream receiver IP network address. Defaults to `127.0.0.1`

`port`

Defines the TUIO stream receiver IP port. Defaults to `3333`

Note: *Config.txt also supports a* `TUIOTracker` *block for use with third party touch screens*. `TUIOTracker` *parameters are not needed when using MultiTaction Cells.* For more information, please contact MultiTaction Support: www.multitaction.com/support-services

C.2 Screen.xml syntax

Screen.xml provides the application computer with details about the display. Specifically, it provides applications with details about where they are displayed on the Cell screens.

A simplified Screen.xml file structure is shown below:

```
<multihead>
   <widthcm>
   <window>
      <area>
         <active>
         <comment>
         <location>
         <size>
         <graphicslocation>
         <graphicssize>
         <keystone>
            <rotations>
            <v1>
            <v2>
            <v3>
            <v4>
         </keystone>
         <seams>
         <colorcorrection>
         <rgbcube>
      </area>
      <location>
      <size>
      <frameless>
      <fullscreen>
      <resizeable>
   </window>
</multihead>
```

These elements are described below and on the following pages.

`<multihead>`

This is the root element. This element contains:

`<widthcm>`

Defines the physical width of the screen in centimeters, as measured with a ruler.

For 55 inch models, the width is 121

The Cornerstone SDK uses the width value for transforming distances in pixels to physical dimensions eg, in gesture detection.

`<window>`

Defines an OpenGL window

`<area>`

Defines an `area` inside the `window`. The `area` corresponds to an OpenGL viewport.

With Cells, you typically only use one large `area`, because there is no need for per-display keystone calibration.

With projection-based systems, each `area` typically corresponds to one projector. Within an `area`, you can do keystone correction and edge-blending for each projector.

`<active>`

Boolean, 1 or 0. Defines whether application content is drawn on this area.

`active` is almost always set to 1 (true).

There are some special cases where non-active areas are useful. Because the computer vision calibration window is drawn also on the inactive `areas`, you can have a separate monitor that you use to adjust the computer vision parameters, but it does not contain any application content.

`<comment>`

Is a text node for adding general comments about the `area`. This is useful when documenting complex setups.

`<location>`

Defines the lower left corner of the `area` relative to the `window`.

`<size>`

Defines width and height of the area. `Size`, together with `location` define the parameters that are passed to the `glViewport` command.

`<graphicslocation>`

Is a 2D vector that specifies the origin for the OpenGL graphics inside the `area`.

If you use just one `area`, this is typically "0 0" (=origin). If you use multiple `areas`, then set up `graphicslocation` to match the graphics coordinates of the top right corner of the `area` to the immediate left of the current `area`.

`<graphicssize>`

Indicates the area used for OpenGL projection.

In practice, the `graphicslocation` and `graphicssize` values are combined and fed to the `gluOrtho` call that is used to set up the bottom-line OpenGL projection.

`<keystone>`

Contains information about keystone correction. It is also used to rotate the graphics output in cases where the Cell (or projector) might be rotated 90°.

`<rotations>`

Indicates the number of rotations that have been applied to the keystone. This is used in the keystone calibration UI to ensure the arrow-buttons lead the keystone correction to the expected direction.

`<v1> <v2 > <v3> <v4>`

These are 2D vectors that contain keystone control point coordinates. The values are in relative terms, so the parameter range is 0-1.

`</keystone>`

`<seams>`

Defines blending areas in projector setups. Contains four numbers that specify the width of the edge-blend in pixels, for each side of the area: left, right, top and bottom.

Background: Old MultiTouch screens are projector-based. To achieve a large uniform display, the projector images must overlap otherwise seams are visible in the intersections. However, the physical overlap region receives roughly twice as much light as the non-overlap regions, resulting in non-uniform brightness. Cornerstone software corrects this by reducing the brightness around the image edges. The size of the edges is defined by the `seams` element.

This element is not needed by newer Cells with LCD screens.

```
        <colorcorrection>
```
A legacy element, used by earlier versions of Cornerstone for color correction. This function is now performed in hardware and `colorcorrection` has been superseded. You do not need to edit this element.

```
        <rgbcube>
```
A legacy element, used by earlier versions of Cornerstone for color correction. This function is now performed in hardware and `rgbcube` has been superseded. You do not need to edit this element.

```
    </area>
    <location>
```
Defines the upper left corner of the `window`.

```
    <size>
```
Defines the width and height of the `window`.

```
    <displaynumber>
```
Specifies the display number in an X Window System on Linux computers.

This element has no effect on Windows computers.

```
    <screennumber>
```
Specifies the screen number in an X Window System on Linux computers.

This element has no effect on Windows computers.

```
    <frameless>
```
Boolean element, 1 or 0. Defines whether the window is drawn without borders.

With `frameless` value 1, the X window manager on Linux is mostly bypassed and the window is rendered on top of everything, regardless whether other 'normal' windows or toolbars want to on top always. Also switching virtual desktops has no effect on frameless rendering.

```
    <fullscreen>
```
Boolean element, 1 or 0. Defines whether the window is drawn in fullscreen mode. This differs from the frameless mode in that the X window manager is not bypassed eg, you can open a terminal window on top of the OpenGL window.

This element has no effect on Windows computers.

```
    <resizeable>
```
Boolean element, 1 or 0. Defines whether the window can be resized.

This is a deprecated feature and is not needed. If the window has borders (ie, `frameless` is 0), the window is resizable anyway.

```
  </window>
</multihead>
```

Appendix D. Firmware change history

Note: *The following list includes both GA and internal-only firmware releases.*

2.0.8-taction21-1

15 Feb 2017

- Use inflow/outflow temperature sensors instead of VM1

2.0.8-taction20-1

18 Apr 2016

- Remove portrait mode
- Disable geometry calibration with touch
- Fixed more issues with Debian Squeeze mirrors
- Add unprocessed image display to System->Maintenance
- Move Geometry calibration inspection under System->Maintenance
- Rename Netmask to Subnet Mask in Setup

2.0.8-taction18

08 Feb 2016

- Clarify TUIO documentation
- Make OSD orientation persistent
- Rename AMX to "Remote Control"
- Don't use squeeze-lts mirrors for apt
- Add documentation for geometry calibration inspection mode
- Merge dgu-detect.py fixes from 1.9.3

2.0.8-taction17

07 Dec 2015

- New touch operated OSD
- Tracker is now its own service MultiTaction-trackerd
- Computer vision improvements
- Portrait mode support
- Power saving mode when display is not showing any image
- Diagnostics over network
- Ability to reset to factory settings
- Remote syslog
- Log viewer
- Brightness slider
- Experimental cell syncing

1.9.3-taction16k

15 Mar 2016

- Fixed TUIO sender error handling, now it can recover from network issues
- Fixed more Squeeze archive.debian.org mirrors

1.9.3-taction16j

10 Mar 2016

- Moved rest of Squeeze LTS repositories to archive.debian.org

1.9.3-taction16i

02 Mar 2016

- Fix updates by using archive.debian.org since Squeeze LTS support ended

1.9.3-taction16h

15 Feb 2016

- Internal changes for production purposes

1.9.3-taction16g

04 Feb 2016

- Internal changes for production purposes

1.9.3-taction16f

15 Jan 2016

- Internal changes for production purposes

1.9.3-taction16e

26 Nov 2015

- Use the new update protocol and server URL

1.9.3-taction16d

04 Sep 2015

- Internal changes for production purposes

1.9.3-taction16c

24 Jul 2015

- Fix updates from old displays

1.9.3-taction16b

09 Jun 2015

- Internal changes for production purposes

1.9.3-taction16

01 Jun 2015

- Added MT555 support
- Removed DRM
- Add ability to read BIOS product name and product version

1.9.2-taction15h

24 Mar 2015

- Fixed external network adapter detection
- Fixed public network interface driver bug on Fujitsu MBs
- Fixed issue with black dots on the picture when using NVIDIA drivers
- Add ability to write EDID with VM1 4.5

1.9.2-taction15f

20 Feb 2015

- Taction setup: Fixed issues with multiple network adapters
- Taction setup: Fixed crash
- Fixed random DHCP / MAC address issue with Sapphire MB

1.9.2-taction15e

06 Feb 2015

- Fixed compatibility issues with Windows 8.1
- Check and fix automatically various settings in Windows on every boot
- Fixed background model sometimes getting darker
- Faster and more robust update downloading
- HTTP proxy support in Windows

1.9.2-taction15d

18 Jun 2014

- Camera re-initialization when needed, fixes startup issues on certain hardware

1.9.2-taction15c

19 May 2014

- Fixed regression bug in start-multitaction.sh

1.9.2-taction15b

13 May 2014

- VM1 serial port connection fixes
- TUIO timetag fix
- Add radius-multiplier to CircleTracker for fine-tuning the real circle radius
- Change min-length-relative default value for square markers

1.9.2-taction15a

27 Nov 2013

- Fixes to tracker XML output when there are network problems
- Correct time tag to TUIO bundles
- No longer show damaged frames in OSD
- Automate firmware updates on Windows, no need to click through installation wizard anymore
- Better default parameters for image merging in UTB devices

1.9.2-taction15

17 Oct 2013

- Fixed an issue with OSD restarting every 1-2 hours when the device is in standby mode
- Fixed a regression issue where large square markers could report incorrect codes
- Added marker velocity and acceleration to TUIO output
- Fixed correct units for velocity and invalid value for acceleration in TUIO finger output
- MultiTaction device licenses no longer expire

1.9.2-taction14f

13 Sep 2013

- Added an option to change the background removal mode

1.9.2-taction14e

05 Sep 2013

- Fixed reboot/shutdown commands from Windows tracker

1.9.2-taction14d

03 Sep 2013

- Tracker background calibration now works correctly if hybrid tracking is enabled

1.9.2-taction14c

22 Aug 2013

- Embed MultiTaction Setup utility to Windows tracker package
- Removed old and unused AutoCalibration.exe
- More robust Windows tracker service launching
- Fixed crash on stand-by mode when having Cornerstone 2.0 clients connected

1.9.2-taction14b

30 Jul 2013

- Site Manager: Fixed auto-calibration state query
- Site Manager: Fixed color-correction VM1 upload

1.9.2-taction14a

12 Jul 2013

- Increased watchdog check intervals due to changed geometry calibration running time

1.9.2-taction14

24 Jun 2013

- Increased temperature limits
- Fixes to temperature monitoring
- Fixed an issue where OSD can hang on exit and can make the unit unresponsive
- Improved marker tracking
- Improved geometry calibration

1.9.2-taction13b

18 Jun 2013

- Increased VM1 temperature limits

1.9.2-taction13a

05 Jun 2013

- Fixed a regression bug in auto-calibration

1.9.2-taction12

24 May 2013

- SiteManager support for embedded devices
- Production tool support
- More robust DGU register read code
- VM1 support for Windows models
- Added temperature thresholds
- Fixed AMX ping -issue
- Windows AMX support
- More robust service start code in Windows models
- Added finger keep-alive -parameter to OSD
- Create a new config file is the old one is missing/invalid

1.9.2-taction11

14 May 2013

- Added uptime statistics to tracker (for production)
- Fixed L-variant update script

1.9.2-taction10

29 Mar 2013

- Support for updating existing devices to use Cuda Tracker
- More robust geometry calibration, especially in Windows
- Limited AMX/SiteManager support for embedded Linux without starting OSD
- Removed conflicts between CornerStone 1.2 and tracker files in Windows
- Color calibration can be toggled on/off in OSD to compare before/after results
- Increased maximum keep alive value for square markers to 100
- Miscellaneous bug fixes

1.9.2-taction9

28 Feb 2013

- Windows support for Cuda Tracker
- AMX 1.1.0 support
- Fixed ghost markers sometimes being left on screen

1.9.2-taction8

17 Jan 2013

- Cuda Tracker
- Temperature monitor
- Key-event forwarding to SiteManager

1.9.2-taction7

11 Dec 2012

- Fixed a marker burn-in issue
- Fixed tracker sometimes crashing when cleaning the screen
- Enhanced background calibration

1.9.2-taction6

27 Nov 2012

- Fixed a regression bug for older FPGA

1.9.2-taction5

26 Nov 2012

- Image brightness hotfix for MT550

1.9.2-taction4

21 Nov 2012

- Hybrid tracking
- Camera register corruption fixes

1.9.2-taction1

19 Oct 2012

- AMX beacon and AMX server
- Standby mode
- New color correction curves / color calibration support

1.9.1-taction16-triangles2

03 Oct 2012

- Improved triangle tracking
- Fixed initial rotation value for square markers

1.9.1-taction16-triangles1

19 Sep 2012

- Triangle tracker support

1.9.1-taction16

25 Jul 2012

- Improved tracking with multiple devices

1.9.1-taction15

17 Jul 2012

- Fixed image issue in OSD

1.9.1-taction14

04 Jul 2012

- Marker tracking improvements for MT550
- Better default preset for circle tracking
- Internal changes

1.9.1-taction13

29 Jun 2012

- IR Pen tracker
- TUIO sender rewrite
  - Remove one sample delay
  - Alive/fseq are always sent only once per bundle, fseq is always the last message
  - Fixed issue that packet size might go over MTU limit when using objects-as-fingers
  - Every bundle is now a separate packet handling only one type of messages
- Beta version of CircleTracker
- PenMarker -> SquareMarker conversion for cornerstone 1.2 clients
- Ignore unknown values when de-serializing configuration for better backwards compatibility (config.xml)
- Twinkle background fix
- OSD: Enable/Disable CV extensions
- OSD: Remove most of the socket error flood
- OSD: Added separate "External Source" and "Quit" -buttons
- OSD: New layout with logo
- Windows: VM1Svc fixes, switching between external and internal input works better
- Windows: Added BinaryData/XML stream to Windows firewall
- Windows: If trying to run tracker in Win7 app, close the service automatically
- Windows: Don't save screen.xml and config.xml to wrong directories
- Tracker: Fixed SquareMarker marker location fractional part truncation
- Tracker: Use unique ids in marker trackers
- Tracker: No more artifacts when changing blending parameters
- Tracker: Added "edge adjust" parameter to the ContourTracker
- Tracker: CV Extension configuration is now better embedded to config.xml
- Tracker: CV Extensions handle damaged frames better

1.9.1-taction12

07 May 2012

- OSD: Show download progress
- OSD: Remove extra sliders
- Deft configuration is now forward compatible
- Bug fixes

1.9.1-taction11

02 May 2012

- MT550L support

- Handle DGU errors without restarting the application
- Fixed crash on shutdown

1.9.1-taction10+pentracking1

26 Apr 2012

- Better pen tracking near edges
- TUIO delay decreased

1.9.1-taction10

19 Apr 2012

- Fixed issue when tracking could freeze with older DGU firmware

1.9.1-taction9+pentracking1

19 Apr 2012

- Pen tracking support
- Color correction publicly enabled
- Better finger orientation detection
- Fixed a bug where markers had wrong rotation when moving
- A number of other computer vision improvements
- Better TacMan support
- TUIO fixes

1.9.1-taction9

12 Apr 2012

- Support MT420 and MT550W7
- Updated all presets and default values
- Fixed OSD timeout and internal/external display selection (VM1 fixes)
- Support for new DGU with different IP address
- Do not allow to use conflicting IP network with external network ("192.168.0.0/24-problem")
- Fixed some rare cases when network settings didn't apply properly
- Color correction publicly enabled
- Better finger orientation detection
- Fixed a bug where markers had wrong rotation when moving
- A number of other computer vision improvements
- Better TacMan support
- TUIO fixes

1.9.1-taction6

15 Feb 2012

- Color-correction with VM1
- Virtual keyboard fixes, including a numeric keyboard for IP addresses
- Optimized style changes, OSD is much faster now
- OSD: IP settings are read-only in DHCP mode
- OSD: Show VM1 version and state information in about-widget
- OSD: Slider improvements
- OSD: Show MAC address in about-screen
- OSD/MultTaction: HTTP Proxy support/UI
- Handle damaged frames better in CV

1.9.1-taction5

20 Jan 2012

- Simple tracking visualization

- Input-transform fixes
- Advanced-mode with sliders

1.9.1-taction4

16 Jan 2012

- Bug fixes
- Presets
- New input handling

1.9.1-taction2

20 Dec 2011

- Deft error handling

1.9.1-taction1

19 Dec 2011

- Bug fixes
- TUIO support
- UI updates

1.9.0-taction7

09 Dec 2011

- Fixed lumilimit default value
- OSD timeout (preparation for VM1)

1.9.0-taction6

05 Dec 2011

- Fixed a bug in fstab-parser

1.9.0-taction5

30 Nov 2011

- Read-only filesystem support
- Fixed OSD configuration saving
- Memory leak fixes

1.9.0-taction3

16 Nov 2011

- Fixed auto-calibration / network-related issues

1.9.0-taction2

15 Nov 2011

- Bug fixes in OSD / auto-calibration

1.9.0-taction1

11 Nov 2011

- Added about widget

1.9.0-taction0

11 Nov 2011

- First MultiTaction build

FCC compliance

Product Name	MULTITACTION
Model number	MT553UTB MT555UTB

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

- This device may not cause harmful interference, and
- this device must accept any interference received, including interference that may cause undesired operation.

MultiTouch Americas, ltd.

101 Church St. Suite 25, Los Gatos, CA, 95030

Phone number: 1-844-403-3070

Radio and television interference

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and the receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Changes and Modifications not expressly approved by the manufacturer or registrant of this equipment can void your authority to operate this equipment under Federal Communications Commission rules.

Warranty service policy and procedures

Definitions

- **Authorized Reseller** — A reseller authorized by MultiTouch to sell the Products.
- **Buyer** — The first purchaser of a Product or Products from MultiTouch or Authorized Reseller.
- **MultiTouch** — MultiTouch Ltd. and its subsidiaries.
- **Product or Products** — Multi-touch displays manufactured by MultiTouch, which are branded as MultiTouch Cell or MultiTaction Cell.

General Terms and Conditions

1. The Products carry a limited warranty for a period of twelve (12) months, applicable from the date of delivery of the Products to the Buyer by MultiTouch or by Authorized Reseller, as the case may be.
2. This limited warranty covers only the Products. It does not cover software or non-MultiTouch Cell- or MultiTaction Cell-branded products.
3. This warranty is valid only when Buyer either presents proof of purchase consisting of original invoice or sales slip indicating the date of purchase, model and serial number of this Product, or properly completes and presents the warranty card, indicating the date of purchase, the name of the Authorized Reseller, the model of this Product, and the serial number of this Product. MultiTouch reserves the right to refuse warranty if this information has been removed, obliterated, or altered after the original purchase of this Product from MultiTouch or the Authorized Reseller, where applicable. Buyer shall notify MultiTouch in writing of the claimed defects and demonstrate to MultiTouch’s satisfaction that said defects are covered by this limited warranty.
4. MultiTouch's obligations are limited to repair of the defect or replacement of the defective part or, at its option, replacement of this Product itself.
5. Only MultiTouch Authorized Resellers may carry out warranty repairs. Warranty is void if any unauthorized service provider has attempted repair. MultiTouch shall not be liable for reimbursements, claims, or damages that may result from any unauthorized repair of this Product.
6. Repair or replacement under the terms of this warranty does not provide Buyer a right to extend or renew the warranty period. Repair or direct replacement of this Product under the terms of this warranty may be fulfilled.
7. The warranty is only applicable to defects in material, design and workmanship of this Product.
8. By way of an example only, and not as a limitation, the warranty does not cover the following, unless specifically authorized in writing by MultiTouch:
   - Periodic checks, maintenance, repair and replacement of parts due to normal wear and tear;
   - Abuse or misuse, including but not limited to the failure to use the Product for its normal purposes or in accordance with MultiTouch's instructions on usage and maintenance;
   - Any damage caused by the combination of this Product with other products;
   - Any damage resulting from the interaction of this Product with any software or hardware, including, but not limited to, loss or corruption of data;
   - Any damage caused to the Product by viruses or other types of malicious code, or malicious intrusion;
   - Failure or malfunction of this Product arising from incorrect or improper installation or use not consistent with the instructions and technical or safety standards prescribed in the user manual associated with this Product;
   - Accidents, acts of nature such as lightning, water, and fire, public disturbances, improper ventilation, voltage fluctuations, or any other cause beyond the reasonable control of MultiTouch;

- Unauthorized modifications of this Product for the purpose of complying with local or international technical standards in countries for which this Product was not originally designed or intended;
- Any of the seals on this Product's enclosure or parts are broken or show evidence of tampering;
- Defects caused to this Product by improper use as determined by MultiTouch;
- Modification or alteration of any nature that is made in the electrical circuitry/ or physical construction of this Product;
- Installation/ repair work on this Product that is carried out by persons/agency other than those authorized by MultiTouch;
- Site (premises where this Product is kept) conditions that do not conform to the recommended operating conditions of this Product;
- Defects occurring while this Product is in transit;
- Damage to and/or removal of the protective glass/plastic of this Product after Buyer has received this Product; or
- When this Product is leased, rented, loaned, or given as promotion, either by MultiTouch or by Buyer.

9. This warranty does not affect either the consumers' statutory rights or the consumers' rights against the Authorized Reseller related to their purchase/sales agreement.
10. This warranty is not transferable or assignable. This warranty will be Buyer's sole and exclusive remedy and neither MultiTouch nor its Authorized Resellers shall be liable for any incidental or consequential damages or breach of any express or implied warranty of this Product. DISCLAIMER: MultiTouch shall not be liable for the loss of any saved/stored data in Products that are either repaired or replaced.
11. This warranty is exclusive. The sole and exclusive obligation of MultiTouch is to repair or replace the defective Product in the manner and for the period provided above. MultiTouch shall not have any other obligation with respect to this Product or any part thereof, whether based on contract, tort, and strict liability or otherwise. Under no circumstances, whether based on this limited warranty or otherwise, shall MultiTouch be liable for incidental, special or consequential damages.
12. Oral or written statements made by MultiTouch's employees or representatives DO NOT CONSTITUTE WARRANTIES, and shall not be relied upon by Buyer, and are not a part of the contract for sale or this limited warranty.

    The above policies are for Product warranty service and Buyer will be responsible for any costs associated with non-warranty conditions. MultiTouch reserves the right to make final decisions regarding problem determination and the appropriate service option. Replacement units assume the remaining warranty of the original Product.

Limited Warranty Statement for MultiTouch Products

1. Only MultiTouch Authorized Resellers may carry out repairs or replacements.
2. In the event of repairs/replacement of any part/s of this Product, this warranty will thereafter continue and remain in force only for the unexpired period of the warranty. Moreover, the time taken for repair/replacement and in transit whether under the warranty or otherwise shall not be excluded from the warranty period.
3. MultiTouch or an Authorized Reseller reserves the right to retain any part/s or component/s replaced at its discretion in the event of a defect detected in this Product during the warranty period.
4. Buyer must communicate any change of address to either MultiTouch or concerned Authorized Reseller in writing for the continuation of warranty.
5. Warranty for the unexpired period shall continue after this Product is inspected by either MultiTouch or the concerned Authorized Reseller, as the case may be, and found free from transit damage.

6. In case of any transit damage, this Product shall be repaired and the repair shall be invoiced by either MultiTouch or the Authorized Reseller in question, as the case may be, and warranty for the unexpired period shall continue.
7. The warranty does not cover the use of the Product(s) for demonstration purposes.
8. The warranty does not cover accessories external to this Product.

The Authorized Reseller with whom this Product is registered for warranty service, or MultiTouch, where applicable, will instruct Buyer as to whether to effect the warranty service on site at Buyer's premises or at the premises of either Authorized Reseller or MultiTouch. If warranty service is effected at the premises of either Authorized Reseller or MultiTouch, warranty service shall be carried out on a "Carry In" basis, wherein Buyer shall, based on the instructions it has received, bring this Product to either Authorized Reseller or MultiTouch. If warranty service is effected at Buyer's premises, Buyer shall pay a visiting charge, including all travel costs such as flights and other transportation, accommodation, daily travel expense allowances and other travel-related costs related to warranty service, as applicable. The warranty does not cover cost of transportation of a Product from place of installation to MultiTouch or the Authorized Reseller, as the case may be.

Limited Warranty Statement Extension Concerning Dead Pixels in LCD Based Products

Each MultiTouch LCD based Product may have a maximum of one (1) non-significant dead pixel in LCD panel after manufacturing. During the warranty period a maximum of six (6) non-significant dead pixels are permitted without the need for any repair or replacement of the LCD panel covered by the warranty.

In the event of repairs/replacement of any part/s of the unit, this warranty will thereafter continue and remain in force only for the unexpired period of the warranty. Moreover, the time taken for repair/replacement and in transit whether under the warranty or otherwise shall not be excluded from the warranty period.

Limited Warranty Statement Extension Concerning Image Retention/Burn-In in LCD Based Products

Damage to MultiTouch LCD-based Products caused by so-called image-retention or image burn-in is NOT covered by this warranty. Please ensure that the precautions in any documentation supplied with the Product are followed in order to avoid permanent damage to it.

Disclaimer

THERE ARE NO EXPRESS WARRANTIES OTHER THAN THOSE LISTED AND DESCRIBED ABOVE, AND NO WARRANTIES WHETHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, SHALL APPLY AFTER THE EXPRESS WARRANTY PERIODS STATED ABOVE, AND NO OTHER EXPRESS WARRANTY OR GUARANTY GIVEN BY ANY PERSON, FIRM OR CORPORATION WITH RESPECT TO THIS PRODUCT SHALL BE BINDING ON MULTITOUCH. MULTITOUCH SHALL NOT BE LIABLE FOR LOSS OF REVENUE, PROFITS, OR DATA, FAILURE TO REALIZE SAVINGS OR OTHER BENEFITS, OR ANY OTHER SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES CAUSED BY THE USE, MISUSE OR INABILITY TO USE THIS PRODUCT, REGARDLESS OF THE LEGAL THEORY ON WHICH THE CLAIM IS BASED, AND EVEN IF MULTITOUCH HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. NOR SHALL RECOVERY OF ANY KIND AGAINST MULTITOUCH BE GREATER IN AMOUNT THAN THE PURCHASE PRICE OF THIS PRODUCT SOLD BY MULTITOUCH AND CAUSING THE ALLEGED DAMAGE. WITHOUT LIMITING THE FOREGOING, BUYER ASSUMES ALL RISK AND LIABILITY FOR LOSS, DAMAGE OR INJURY TO BUYER AND PURCHASER'S PROPERTY AND TO OTHERS AND THEIR PROPERTY ARISING OUT OF THE USE, MISUSE OR INABILITY TO USE THIS PRODUCT SOLD BY MULTITOUCH NOT CAUSED DIRECTLY BY THE NEGLIGENCE OF MULTITOUCH. THIS LIMITED WARRANTY SHALL NOT EXTEND TO ANYONE OTHER THAN THE ORIGINAL BUYER OF THIS PRODUCT, IS NONTRANSFERABLE AND STATES YOUR EXCLUSIVE REMEDY.