NEW MEDIA &
COMMUNICATION
TECHNOLOGY

LEAP MOTION

NEW MEDIA

https://www.youtube.com/watch?v=u58mK55mEls

SETUP

https://www.leapmotion.com/setup

(= driver + Airspace tool)

Airspace:

Demo

> visualizer

> app store

> apps

online: https://airspace.leapmotion.com/

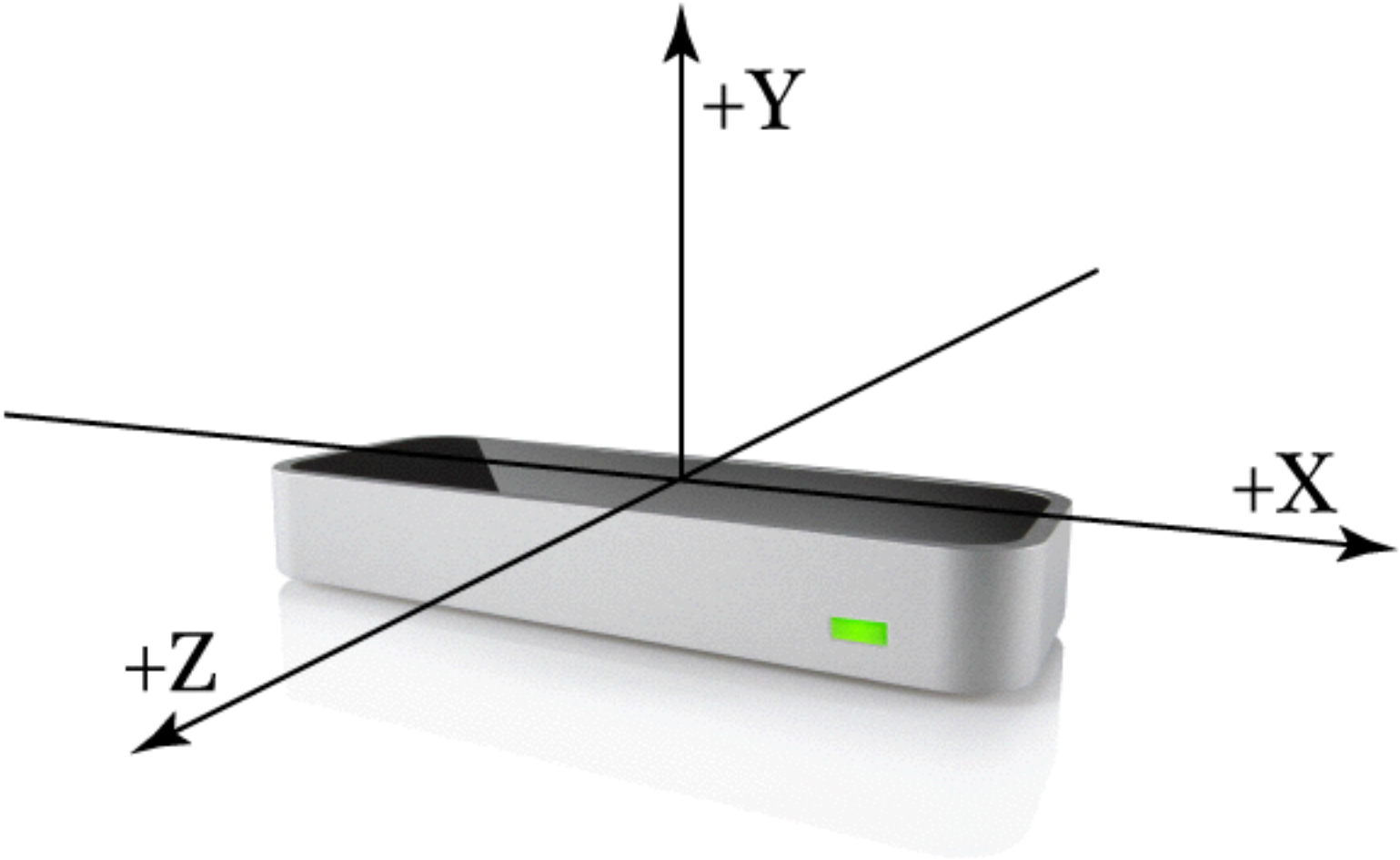
NEW MEDIA &
COMMUNICATION
TECHNOLOGY

WERKING

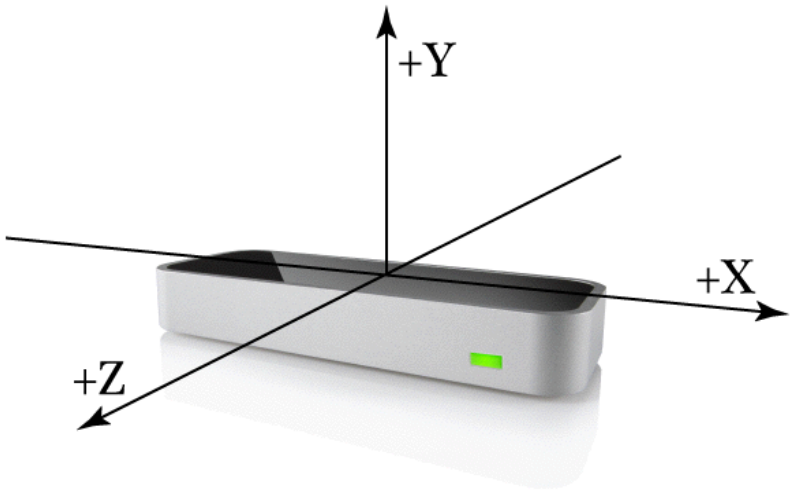
Werking

https://developer.leapmotion.com/

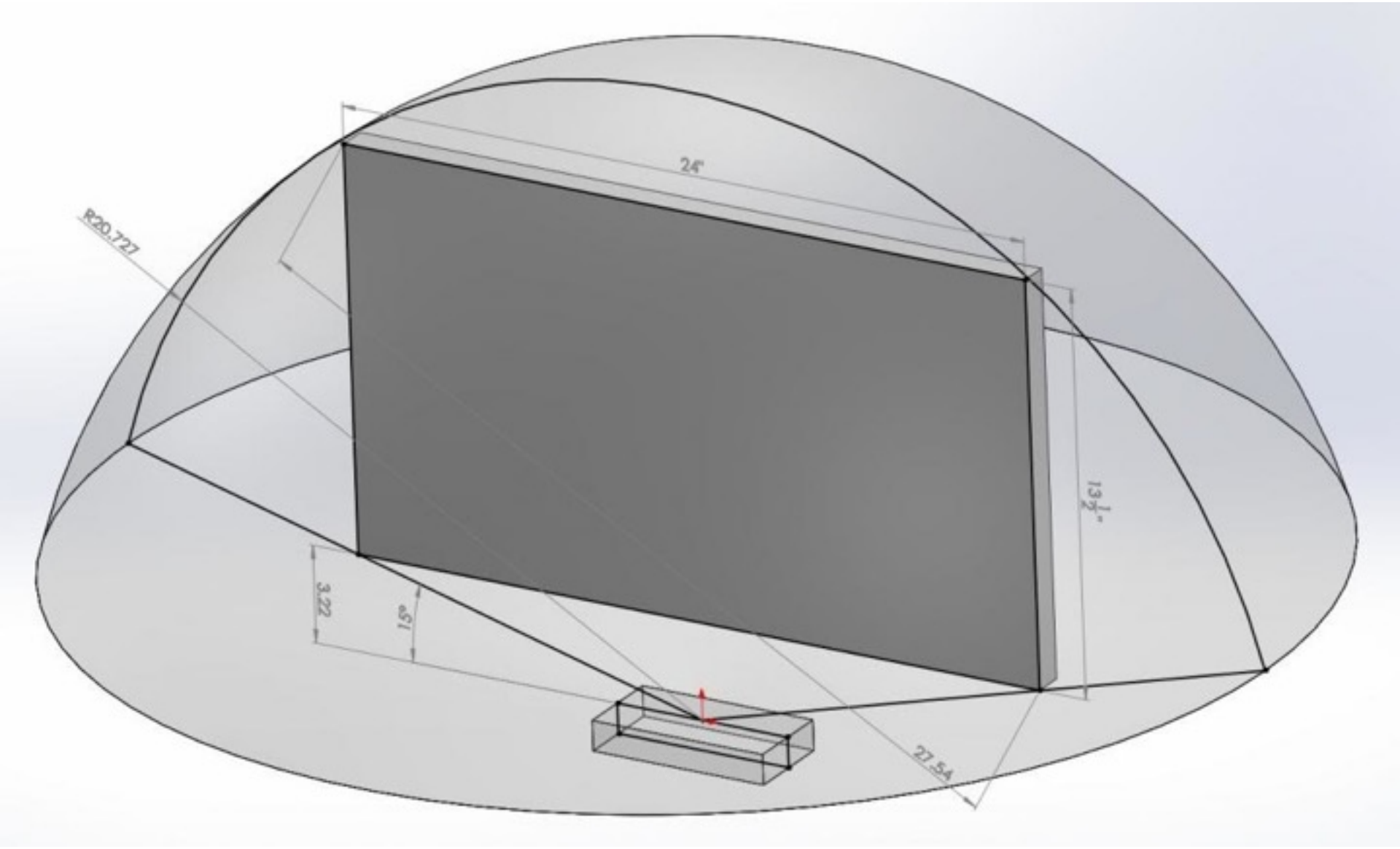
Werking



Werking



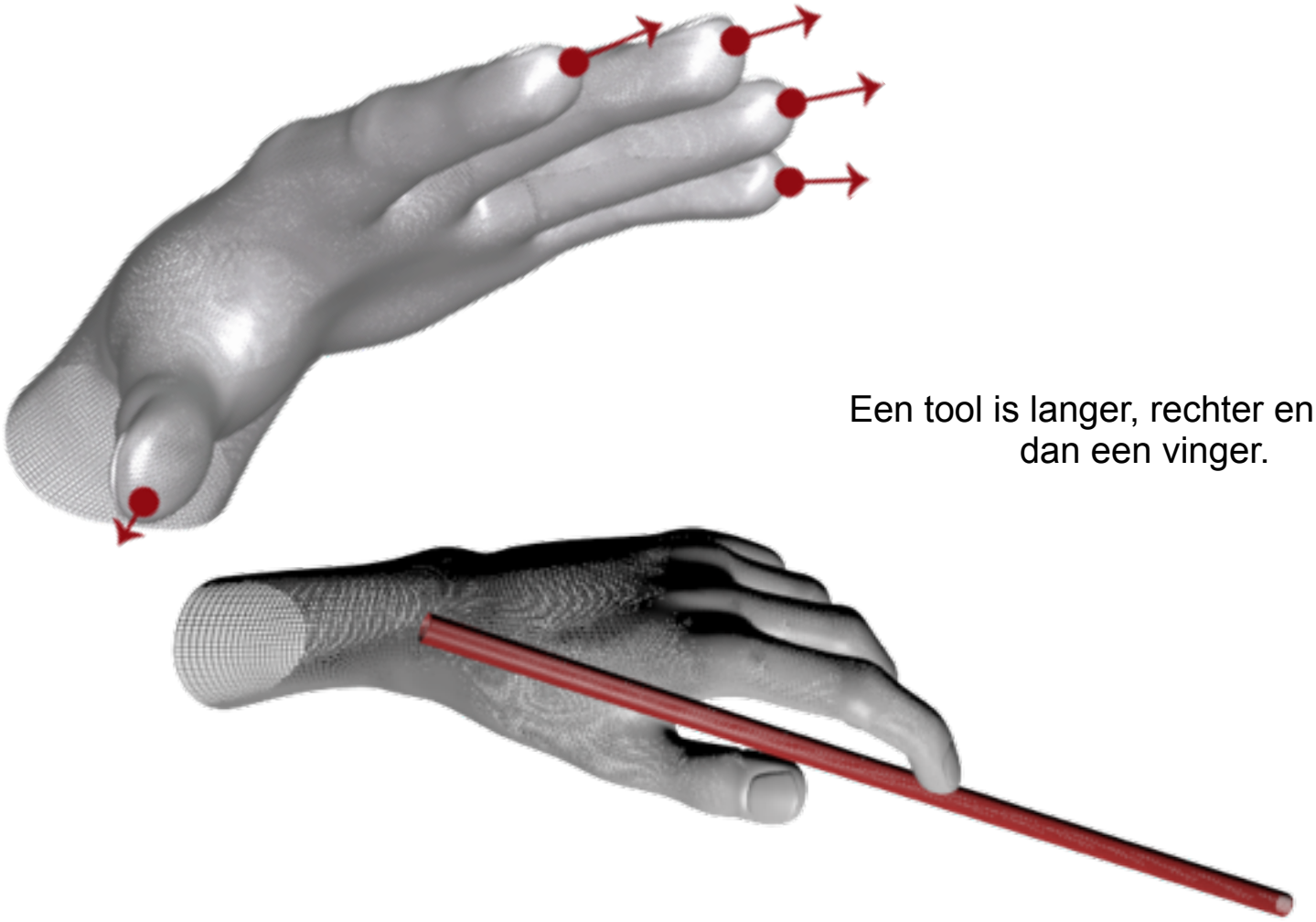
Distance:	millimeters
Time:	microseconds (unless otherwise noted)
Speed:	millimeters/second
Angle:	radians


24"
R20.727
13 1/2"
3.22
15"
27.5A

All about frames

Motion tracking data

Hand, fingers & tools



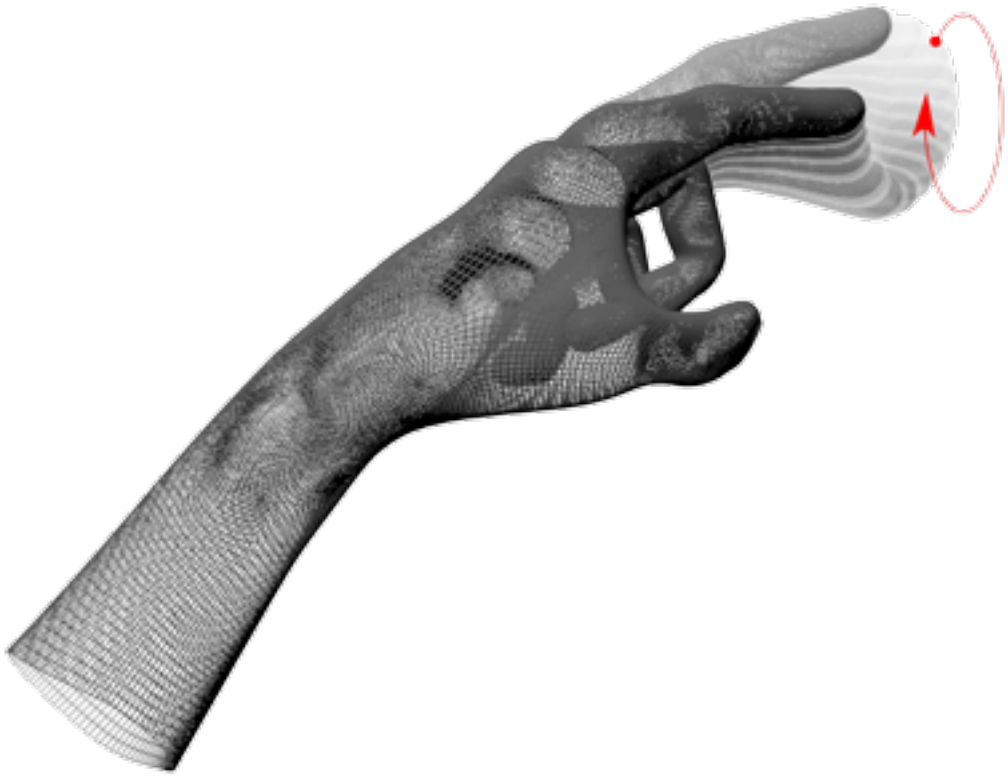
Een tool is langer, rechter en dunner dan een vinger.

NEW MEDIA &
COMMUNICATION
TECHNOLOGY

GESTURES

Circle

cirkel tekenen met bepaalde diameter



Swipe

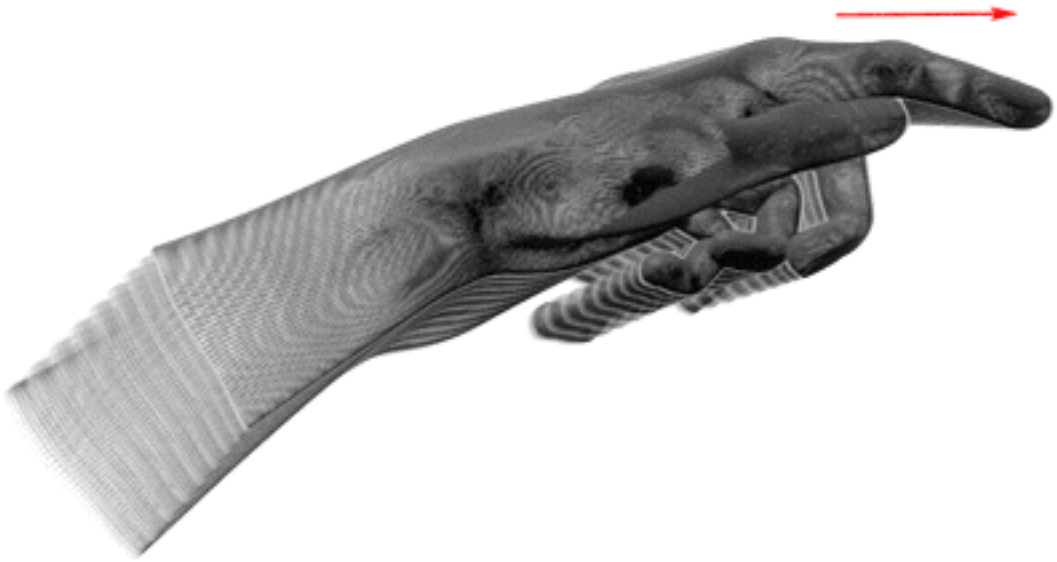
lange lineaire beweging

Key Tap

zoals het aanslaan van een toets

Screen Tap

tikken op een verticaal vlak

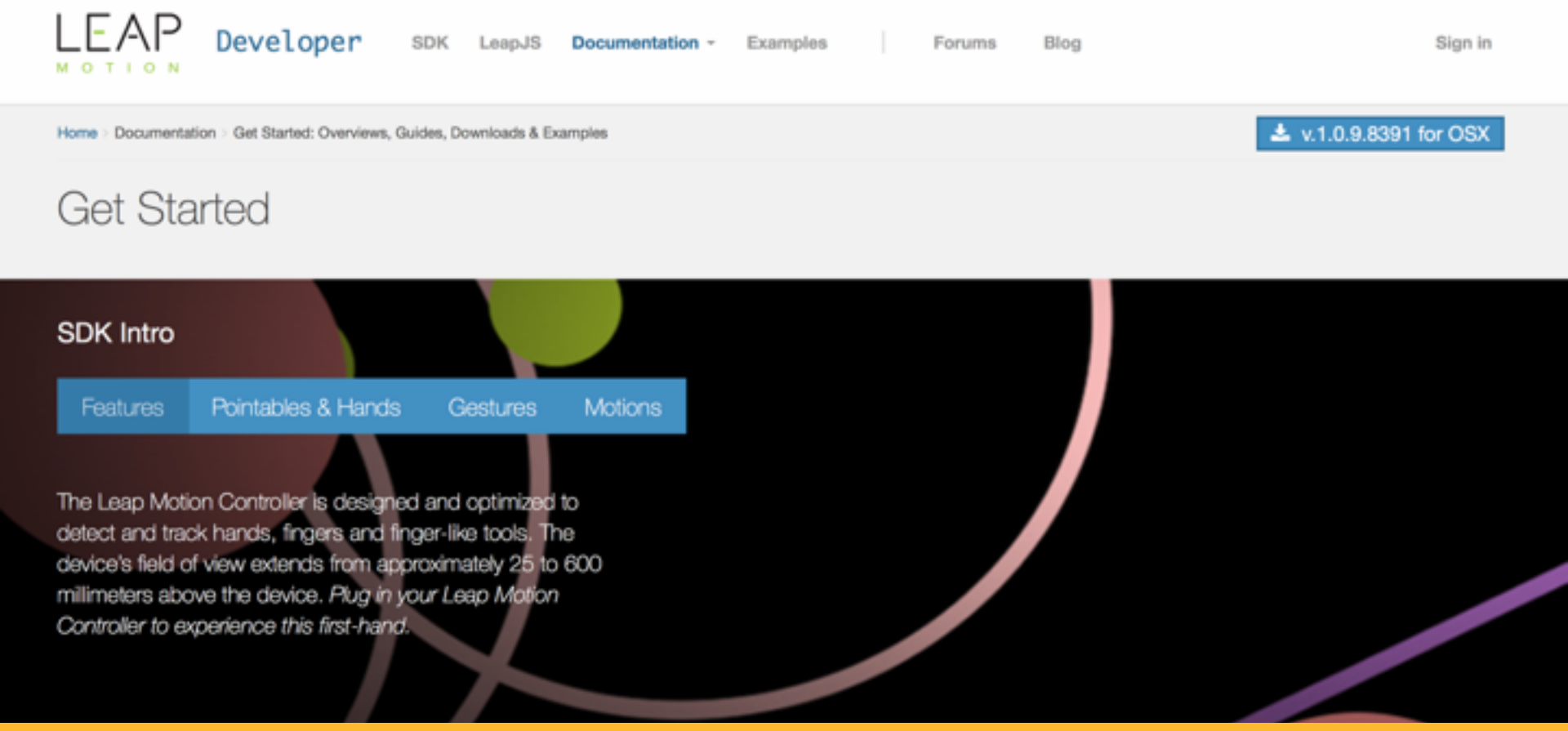


NEW MEDIA &
COMMUNICATION
TECHNOLOGY

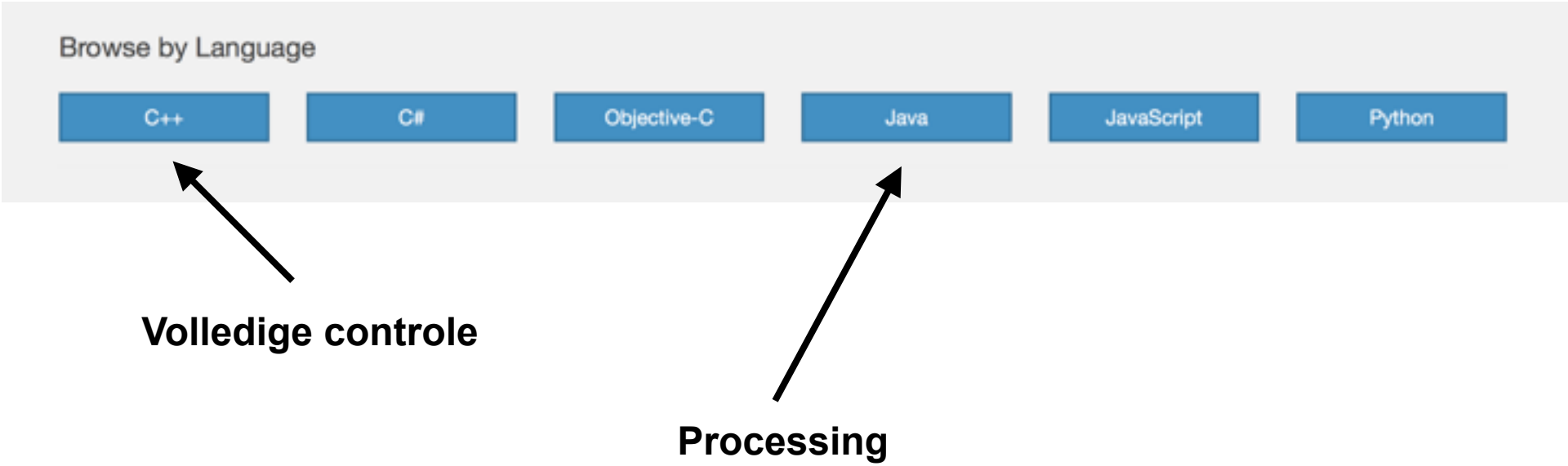
DEV

Documentation

https://developer.leapmotion.com/documentation



Languages



Volledige controle

Processing

NEW MEDIA &
COMMUNICATION
TECHNOLOGY

PROCESSING

Leap Motion

Verschillende libraries

LeapMotion
by Michael Heuer
Forwards Leap Motion controller events to a Processing sketch.

Leap Motion for Processing
by Darius Morawiec
Simple library to use the complete Leap Motion API in Processing.

LeapMotionP5
by informative
Leap Motion library for Processing

Wij gebruiken:

LeapMotion
by Michael Heuer
Forwards Leap Motion controller events to a Processing sketch.

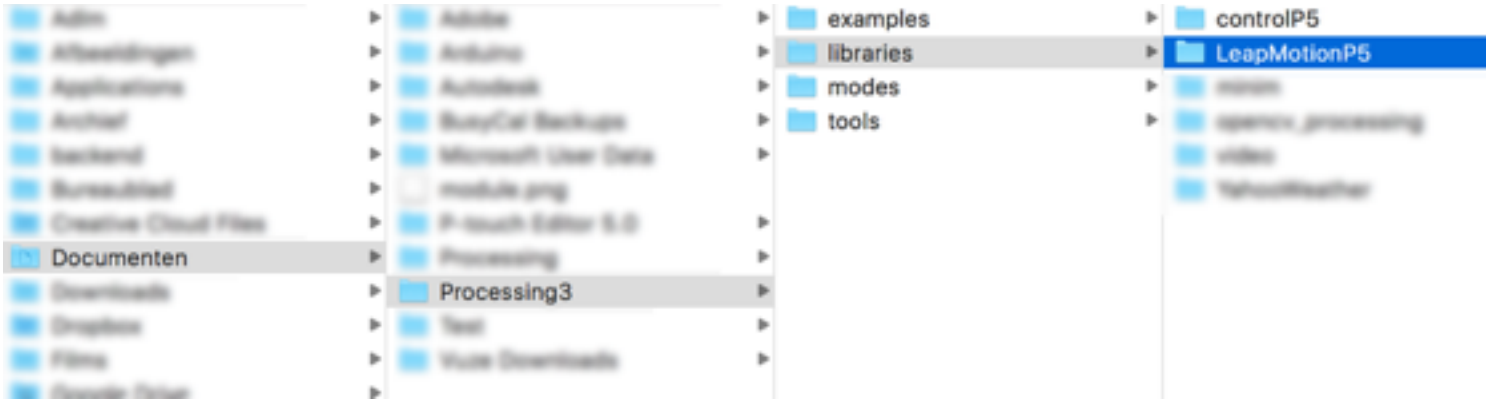
Leap Motion for Processing
by Darius Morawiec
Simple library to use the complete Leap Motion API in Processing.

LeapMotionP5
by informative
Leap Motion library for Processing

Installeren:

LeapMotionP5
by informative
Leap Motion library for Processing

Uitgepakte library kopiëren naar juiste folder



Leap Motion library for Processing

2013

The Leap Motion Controller is a small device tracking the position of your fingers in 3d space. With LeapMotionP5 you can access and use that data in processing.

Since we received the Leap SDK, we had the chance to develop different kind of applications for free fingertip interaction in mid air. To touch, to navigate and even form objects. Unsurprisingly we found speed, feedback and precision delivered by the leap absolutely convincing.
In the course of those experiments, we developed a Processing library which is designed to give you the most easy access to the Leap Motion

List of all functions of the library:

LeapMotionP5(**PApplet** parent);	The constructor
void enableGesture(**Type** gestureType);	Enables a Type of gesture for being recognized
void disableGesture(**Type** gestureType);	Disables a Type of gesture from being recognized
boolean isEnabled(**Type** gestureType);	Returns if a Type of gesture is currently enabled or disabled
Fingers/Tools:	
ArrayList getFingerList();	ArrayList containing all currently tracked fingers
ArrayList getFingerList(**Hand** hand);	ArrayList containing all currently tracked fingers on the hand
ArrayList getFingerList(**Frame** frame);	ArrayList containing all currently tracked fringers in the frame
ArrayList getToolList();	ArrayList containing all currently tracked tools
ArrayList getToolList(**Hand** hand);	ArrayList containing all currently tracked tools on the hand
ArrayList getToolList(**Frame** frame);	ArrayList containing all currently tracked tools in the frame
ArrayList getPointableList();	ArrayList containing all currently tracked pointables
ArrayList getPointableList(**Hand** hand);	ArrayList containing all currently tracked pointables on the hand
ArrayList getPointableList(**Frame** frame);	ArrayList containing all currently tracked pointables in the frame
Finger getFinger(**int** fingerNr);	Returns a Finger or Tool by *nr*. The Pointable which was detected by
Tool getTool(**int** toolNr);	the Leap controller first is *nr* 0, the one detected secondly is nr 1 and
Pointable getPointable(**int** pointableNr);	so forth
PVector getTip(**Tool/Finger** pointable);	Position of the fingertip mapped to the size of the sketch window
PVector getOrigin(**Tool/Finger** pointable);	Position of where the finger 'grows' out of the hand
PVector getVelocity(**Tool/Finger** pointable);	Velocity of the fingertip
PVector getAcceleration(**Tool/Finger** pointable);	Acceleration of the fingertip
PVector getDirection(**Tool/Finger** pointable);	Normalized PVector of the direction the finger is pointing at
PVector getLength(**Tool/Finger** pointable);	Length of the finger
PVector getWidth(**Tool/Finger** pointable);	Width/thickness of the finger
Hand:	
ArrayList getHandList();	ArrayList containing all currently tracked hands
ArrayList getHandList(**Frame** frame);	ArrayList containing all currently tracked hands in the passed frame
Hand getHand(**int** handNr);	Returns a Hand by a number The hand which was tracked by the
	leap first has the nr 0 and the secondly tracked hand has nr 1 etc.
PVector getPosition(**Hand** hand);	Average position of the hand palm
PVector getVelocity(**Hand** hand);	Velocity of the average position of the hand palm
PVector getAcceleration(**Hand** hand);	Acceleration of the average position of the hand palm
PVector getNormal(**Hand** hand);	Normal of the hand palm
PVector getDirection(**Hand** hand);	Normalized direction the hand is pointing at
PVector getSphereCenter(**Hand** hand);	Center of the sphere in your hand
float getSphereRadius(**Hand** hand);	Radius of the sphere in your hand
float getPitch(**Hand** hand);	Pitch rotation of the hand
float getRoll(**Hand** hand);	Roll rotation of the hand
float getYaw(**Hand** hand);	Yaw rotation of the hand

Air Piano
with Leap Motion

DOFL Y. H. YUN
thedofl.com

NEW MEDIA &
COMMUNICATION
TECHNOLOGY

HOW TO LEAP..

Fingers…

```
01  import com.onformative.leap.*;
02  import com.leapmotion.leap.*;
03
04  LeapMotionP5 leap;
05
06  void setup() {
07    size(600, 600, P3D);
08    noFill();
09    stroke(255);
10    leap = new LeapMotionP5(this);
11  }
12
13  void draw() {
14    background(0);
15    for (Finger f : leap.getFingerList()) {
16      PVector position = leap.getTip(f);
17      PVector velocity = leap.getVelocity(f);
18      ellipse(position.x, position.y, 10, 10);
19      line(position.x, position.y, position.x + velocity.x, position.y + velocity.y);
20  }
```

Tools & hand…

ArrayList **getFingerList**();
ArrayList **getFingerList**(Hand hand);

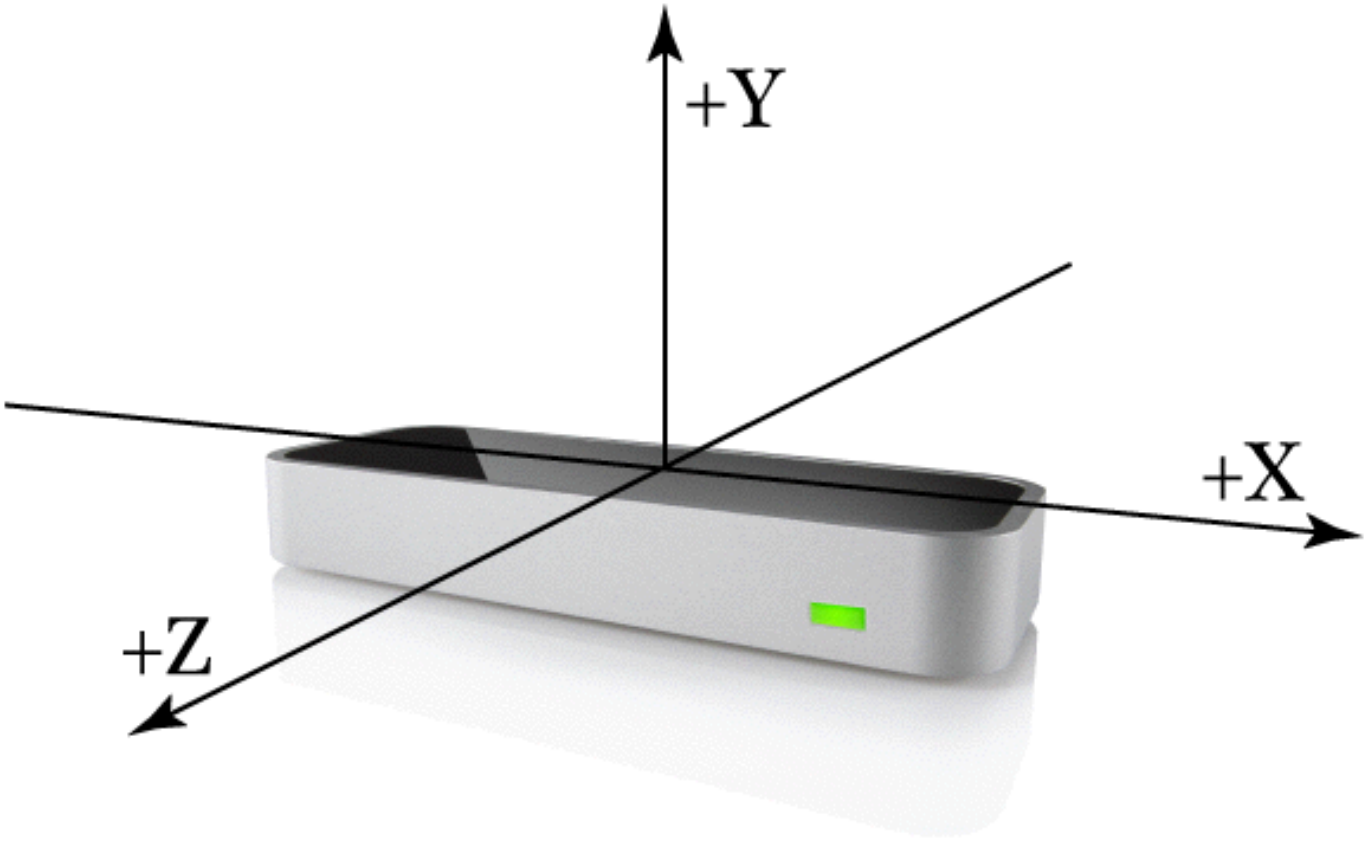
ArrayList **getToolList**();
ArrayList **getToolList**(Hand hand);

ArrayList **getHandList**();

Vectors (finger)

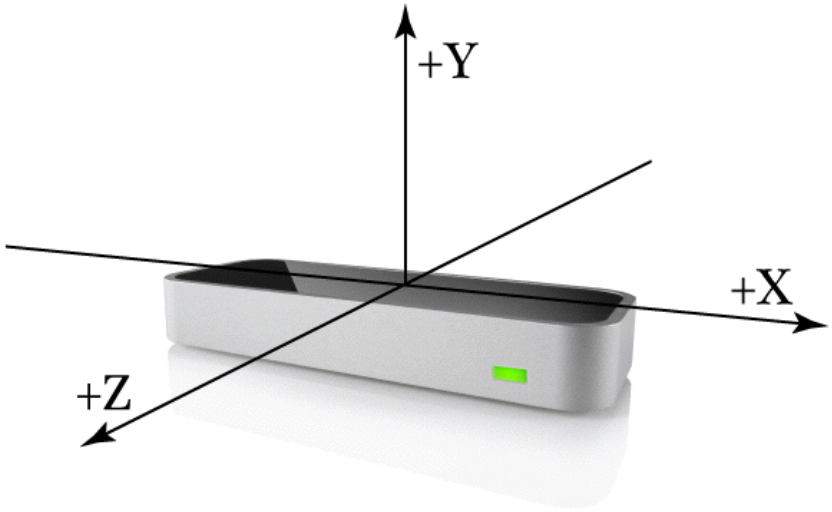
PVector **getTip**(Tool/Finger pointable);	Position of the fingertip mapped to the size of the sketch window
PVector **getOrigin**(Tool/Finger pointable);	Position of where the finger 'grows' out of the hand
PVector **getVelocity**(Tool/Finger pointable);	Velocity of the fingertip
PVector **getAcceleration**(Tool/Finger pointable);	Acceleration of the fingertip
PVector **getDirection**(Tool/Finger pointable);	Normalized PVector of the direction the finger is pointing at
PVector **getLength**(Tool/Finger pointable);	Length of the finger
PVector **getWidth**(Tool/Finger pointable);	Width/thickness of the finger

Vector



Vector

Vector([x, y, z])



The Vector struct represents a **three-component mathematical vector** or point such as a direction or position in three-dimensional space.

The Leap Motion software employs a **right-handed Cartesian coordinate system**. Values given are in units of real-world millimeters. The origin is centered at the center of the Leap Motion Controller. The x- and z-axes lie in the horizontal plane, with the x-axis running parallel to the long edge of the device. The y-axis is vertical, with positive values increasing upwards (in contrast to the downward orientation of most computer graphics coordinate systems). The z-axis has positive values increasing away from the computer screen.

Vectors

Vector omzetten naar Processing Vector
PVector via:

leap.**vectorToPVector**(vector);

Gestures

Circle, Swipe, Screentap, Keytap

```
import com.leapmotion.leap.ScreenTapGesture;


leap.enableGesture(Type.TYPE_SCREEN_TAP);
leap.disableGesture(Type.TYPE_SCREEN_TAP);


public void screenTapGestureRecognized(ScreenTapGesture gesture) {
  if (gesture.state() == State.STATE_STOP) {
  }
  else if (gesture.state() == State.STATE_START) {
  }
  else if (gesture.state() == State.STATE_UPDATE) {
  }
}
```

Gestures

Eigenschappen opvragen

Type	gesture.type();
ID	gesture.id();
Position	gesture.position();
Direction	gesture.direction();
Duration	gesture.durationSeconds();

Circle

Radius	gesture.radius();
Clockwiseness	clockwiseness;
Turns	gesture.progress();
Center	leap.vectorToPVector(gesture.center());