NON TACTILE

eine bewegungssensitive Benutzerschnittstelle

Tim Gatzky · Diplompräsentation · 17.01.2012

Gliederung

- Ausgangssituation
- Moodboard
- Intention und Motivation
- Anwendungsgebiete und Gebrauchsszenarios
- Zielsetzung und Aufgabenstellung
- Mensch-Maschine-Schnittstelle
- Recherche

THEORIE

PRAXIS

- Designbriefing
- Entwicklung, Konzeption und Testumgebungen
- Realisierung
- Anwendungsbeispiel
- Prototyp

Ausgangssituation / State of the art

- Computergestützte Systeme gehören zu unserem alltäglichen Leben
- Benutzerschnittstellen wie Touchscreens und Sprachsteuerungen ermöglichen in einer Vielzahl technischer Geräte eine natürliche und intuitive Interaktion

- Technische Geräte sind blinde Gesprächspartner, die nicht auf den Benutzer reagieren wie es ein menschliches Gegenüber tun würde.
- Mensch und Computer begegnen sich nicht auf Augenhöhe, eine wechselseitige Kommunikation ist nicht möglich.

„Wir werden nie aufhören schöne Momente zu teilen, oder in einem guten Buch zu versinken.
Wir werden immer für Freunde kochen und unsere Lieblingsmannschaft anfeuern.
Wir werden weiter an Meetings teilnehmen, unsere Erinnerungen festhalten und neue Dinge lernen,
aber wie wir all das machen, wird nie wieder so sein wie es war.“

Zitat: Apple iPad2 Tv-Werbung

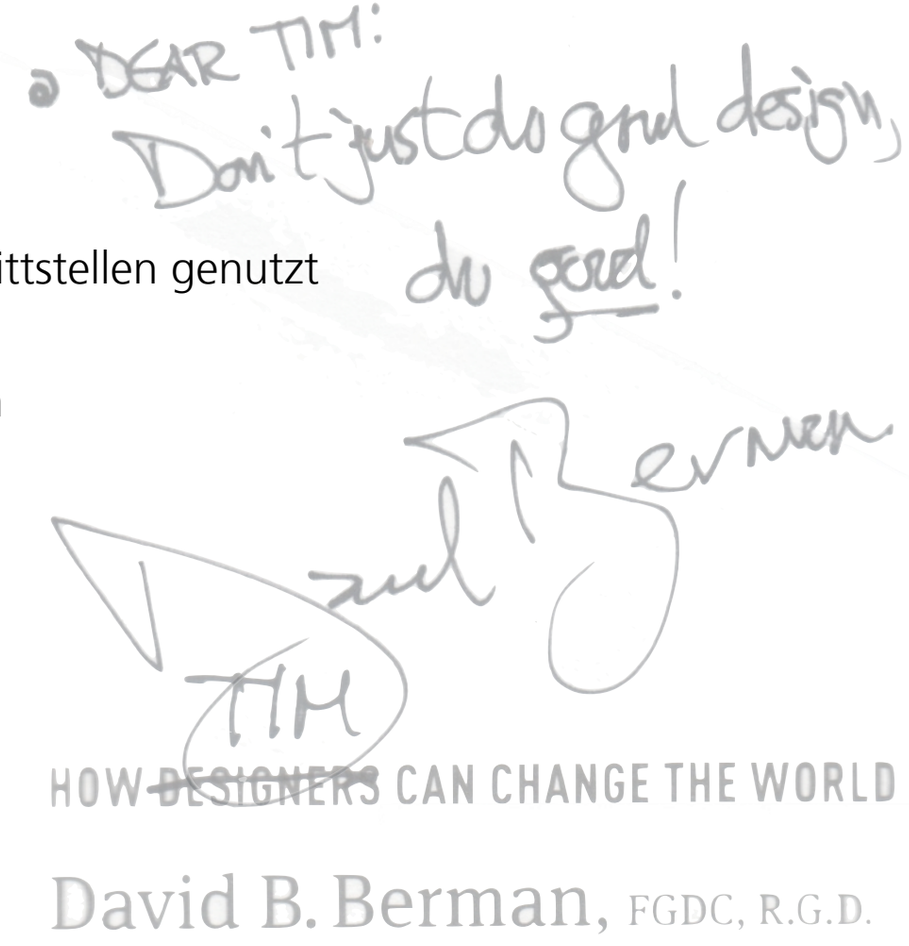
„Wagen wir einen Blick in die Zukunft...“

Moodboard

Intention und Motivation

Der Computer muss sein Gegenüber sehen und deuten lernen. Nicht nur um zwischen ungewollten und nicht auf ihn bezogenen Aktionen zu reagieren bzw. nicht zu reagieren, sondern auch, um dem menschlichen Gegenüber ein angemessener Mitspieler und nicht Gegenspieler zu sein.

- Berührungsfreie Interaktion mit technischen Geräten in Form von natürlichen und intuitiven Körpergesten
- Eine Benutzerschnittstelle, die zusätzlich zu den gewohnten Benutzerschnittstellen genutzt werden kann und diese nicht ersetzt oder in ihrer Funktion beeinträchtigt
- Herabsetzung der Zugangsbarrieren von technischen Geräten und Medien
- Verbesserung der Kommunikation zwischen Mensch und Maschine



Anwendungsgebiete, Gebrauchsszenarios

- Barrierefreier Zugang zu technischen Geräten
- Ökonomischer Gebrauch von technischen Geräten, Eco-Design
- Intelligente Kommunikation mit technischen Geräten, intelligente Alltagsgegenstände
- Hygiene
- Vandalismus
- Interaktive Informationsbereitstellung (z.B. im öffentlichen Raum)
- Bequemeres Leben, höherer Lebensstandard
- Altersgerechtes Benutzen von technischen Geräten (jung und alt)
- Spielerisches Benutzen und Interagieren von/mit technischen Geräten
- Zugang zu technischen Geräten auf Distanz (Gefährliche Bereiche (heiß, kalt))
- Früherkennung von Gefahrensituationen, Fahrerassistanz (z.B. Automobilindustrie)
- uvm...

Aufgabenstellung und Zielsetzung

- Ausgehend von dem Wunsch, Maschinen intelligenter zu machen und dem Menschen ein angemessener Kommunikationspartner zu sein, sollte eine alternative Benutzerschnittstelle entwickelt werden.
- Die Schnittstelle soll als zusätzliche Eingabe-/ Interaktionsmöglichkeit_für den Benutzer bereitstehen.
- Durch ein berühungsfreies und gestengesteuertes Interface soll der Benutzer mit dem Computer kommunizieren können.
- Die Maschine soll direkt darauf reagieren, ob ein Mensch sie anschaut und interagieren möchte.
- (Techn.) Die Interaktion soll ohne zusätzliche Hardware nur mit Standardkomponenten möglich sein um ein möglichst breites Nutzerfeld einzuschließen.

Theoretische Aspekte der

Mensch-Maschine-Schnittstelle

- ...ist: Kontaktstelle zwischen Maschine und benutzendem Menschen
- ...ist: Wesentlicher Bestandteil jeder Interaktion mit einem technischen Gerät

Was zeichnet eine *gute* Benutzerschnittstelle aus?

- Einfachheit
- Das Gefühl von Sicherheit und Kontrolle muss jeder Zeit gegeben sein.
- Auf die Art der Benutzung hin ausgerichtet
- *Feedback* (Rückmeldung vom System) steht in direktem Bezug zur Eingabe
- Benutzerfreundlichkeit
  - Effizient
  - Grundsicherheit des Nutzers und des Produktes
  - Gebrauchswert
  - Leicht zu erlernen wieder zu nutzen

Recherche und vergleichende Analyse

- Amnesty International · *Es passiert, wenn niemand hinsieht!*
- Amnesty International · *Light the dark*
- Webcam Spiele

Recherche

Amnesty Int. · Es passiert, wenn niemand hinsieht!

Eine Kampagne gegen häusliche Gewalt.

...

Eine interaktives Plakat:

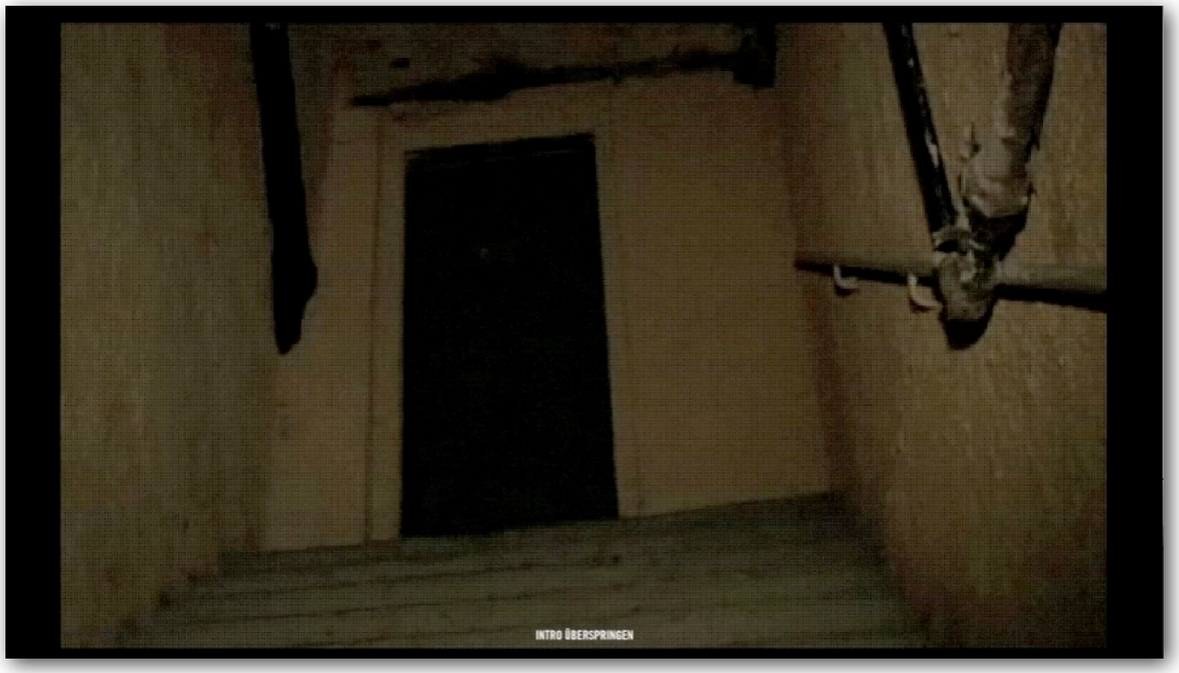
- Wendet sich der Betrachter von dem Plakat ab, werden Bilder häuslicher Gewalttaten gezeigt.
- Dreht sich der Betrachter zum Plakat, wird ein friedliches Beisammen dargestellt.

...

Agentur: Jung van Matt und Wall AG

Recherche

Amnesty International · Light the dark



STOPPEN SIE FOLTER:
BRINGEN SIE LICHT INS DUNKEL!

Eine interaktive Online-Kampagne für die internationale Bekämpfung von Folterungen.

...

- Der Benutzer benötigt eine Lichtquelle um einen dunklen Kellerraum zu durchleuchten.
- Die Webcam des Computers reagiert auf die Lichtquelle und Bereiche des Kellers werden sichtbar bis der virtuelle Folterer verscheucht ist.

...

Agentur: Saatchi & Saatchi

Quelle: http://www.light-the-dark.org


INTRO ÜBERSPRINGEN

Recherche

Online Webcam-Spiele

MotionBubbles

Quelle: http://www.lorenzgames.com

WindowCleaner

Quelle: http://www.lorenzgames.com

Designbriefing, Anforderungsliste

Was muss ich beachten...?

- Einfachheit
- Wiederverwendbarkeit, Flexibilität, Erweiterbarkeit ➜ modularer Aufbau
- Sinnigkeit in der Bedienung und Benutzerfreundlichkeit
- Schnelligkeit und sicheres Feedback
- Im Internet einsatzbar

Was brauche ich...?

- Kopferkennung
- Augenerkennung und Erkennung von Augenaufschlägen
- Bewegungserkennung
- Interpretation von Körpergesten

Was habe ich...?

- Webcam und Laptop

Praktischer Teil

- Entwicklung, Konzeption, Testumgebungen
- Realisierung
- Beispielanwendung
- Prototyp

Entwicklung, Konzeption, Tests Tests Tests...

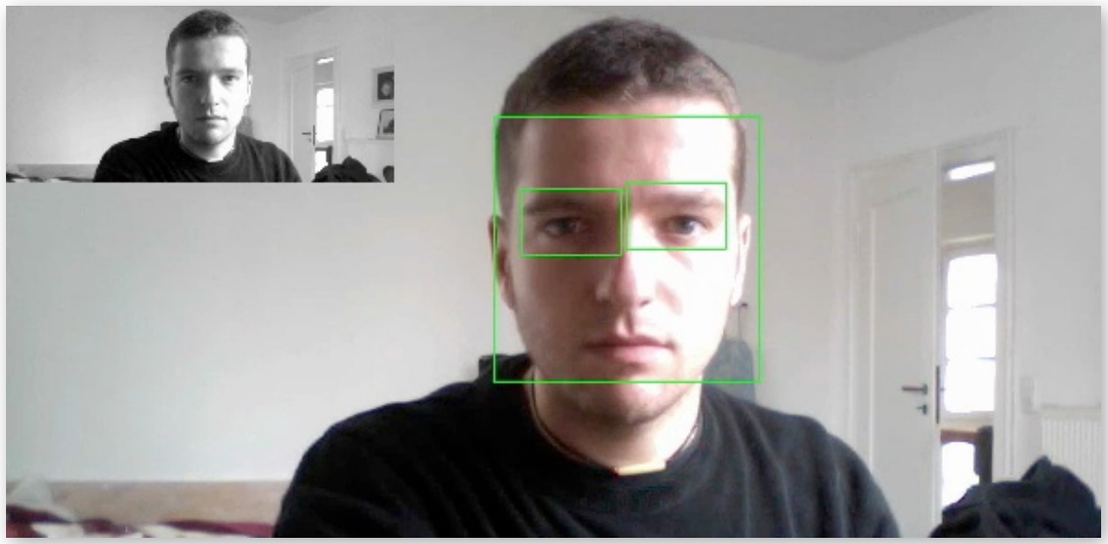
Was brauche ich...?

- Kopferkennung
- Augenerkennung und Erkennung von Augenaufschlägen
- Bewegungserkennung
- Interpretation von Körpergesten

Realisierung

Augenerkennung

- Voraussetzung:
  Finden des Kopfes/Gesichts und der Augen mit Hilfe der Viola-Jones Methode
- Resultat: Ein rechteckiger Bereich der Regionen
- Vorteile: Suchbild kann skaliert werden.



Realisierung

Augenerkennung

```
326⊕    /**
329⊖    private function onRender(e:CameraEvent = null):void
330     {
331         // var clipRect:Rectangle = new Rectangle();
332         camData.draw( cam.bitmapData, null, null, null, null, false );
333
334         // Head and eye tracking
335         var faceRects:Vector.<Rectangle>;
336         var irisPoints:Vector.<Point> = new Vector.<Point>;
337         var ignoreRect:Rectangle = new Rectangle();
338
339         if(this.isTracking)
340         {
341             // Start detection cycle
342             detector.nextFrame();
343             if(detector.state == detector.numSteps - 1)
344             {
345                 faceRects = detector.result;
346                 faceRects = detector.groupRectangles( faceRects );
347
348                 if(faceRects.length)
349                 {
350                     ignoreRect = faceRects[0];
351                     faceRect = new Rectangle( faceRects[0].x * scaleFactor, faceRects[0].y * scaleFactor, faceRects[0].width * scaleFactor, faceRects[0].height *
352
353                     // add event listener here for head
354                     this.dispatchEvent( new TrackingEvent( TrackingEvent.HEAD_FOUND ) );
355
356                     drawRect( faceRects[0], _faceRectContainer, scaleFactor, true, 0xffffff );
357
358                     // drawCircle(new Point(faceRects[0].x + faceRects[0].width/2, faceRects[0].y + faceRects[0].height/2 ), faceRects[0].width/2, faceRectContain
359
360                     // Find Iris
361                     irisPoints = findEyesAndIris( faceRects[0] );
362
363                     if(irisPoints.length)
364                     {
365                         // drawCircles(irisPoints, irisRadius/eyeScaleFactor, irisContainer, 1);
366                         drawCross( irisPoints, 7, _irisContainer, 1, true, 0xffffff );
```

Gesichtserkennung erfolgreich

Berichte, dass Kopf gefunden wurde

Zeichne Rechteck um den Kopfbereich, als visuelles Feedback

Suche Augen und Iris im Kopfbereich

Klassenpfad: com.tgatzky.Interface.as

```as3
/**
 * Find eyes and start iris detection if found
 */
protected function findEyesAndIris(r:Rectangle):Vector.<Point>
{
    var irisPoints:Vector.<Point> = new Vector.<Point>;
    var iris_p:Point = new Point();
    // var eyeRects:Vector.<Rectangle>;
    var eyes_r:Rectangle = r.clone();
    eyes_r.height *= 0.5;
    eyes_r.width *= 0.5;
    eyes_r.y += eyes_r.height * 0.55;

    var eyes_left:Vector.<Rectangle> = new Vector.<Rectangle>;
    var eyes_right:Vector.<Rectangle> = new Vector.<Rectangle>;

    var offset:int = 12;

    // left eye
    eyes_r.x += offset;
    eyes_left = detectorLE.detect( eyes_r, 1, 1.18, 0.05, -1, detector );
    eyes_left = detectorLE.groupRectangles( eyes_left );
    if(eyes_left.length)
    {
        this.dispatchEvent( new TrackingEvent( TrackingEvent.EYE_FOUND_LEFT ) );
        // drawRects(eyes_left, faceRectContainer, scaleFactor, true);
        iris_p = detectIris( eyes_left[0], irisDetectorLE );
        // eyeRects_fallback = eyes_left;
        if(iris_p != null)
        {
            irisPoints.push( iris_p );
            this.dispatchEvent( new TrackingEvent( TrackingEvent.IRIS_FOUND_LEFT ) );

        }
        else
        {
            this.dispatchEvent( new TrackingEvent( TrackingEvent.IRIS_LOST_LEFT ) );

        }
    }
    else
    {
        this.dispatchEvent( new TrackingEvent( TrackingEvent.EYE_LOST_LEFT ) );

    }

    // right eye
    eyes_r.x -= offset;
    eyes_r.x += eyes_r.width;
    eyes_right = detectorRE.detect( eyes_r, 1, 1.18, 0.05, -1, detector );
    eyes_right = detectorRE.groupRectangles( eyes_right );

    if(eyes_right.length)
    {
        this.dispatchEvent( new TrackingEvent( TrackingEvent.EYE_FOUND_RIGHT ) );
        // drawRects(eyes_right, faceRectContainer, scaleFactor,true);
        iris_p = detectIris( eyes_right[0], irisDetectorRE );
        // eyeRects_fallback = eyes_right;
        if(iris_p != null)
        {
            irisPoints.push( iris_p );
            this.dispatchEvent( new TrackingEvent( TrackingEvent.IRIS_FOUND_RIGHT ) );
        }
        else
        {
```

Augenregion beschneiden

Im linken Augenbereich nach Auge suchen

Auge erfolgreich gefunden

Berichte, dass linkes Auge gefunden wurde

Starte Suche nach Iris in dem Bereich

Klassenpfad: com.tgatzky.Interface.as

Interface.as

```as
        /**
         * Locate Iris via BlobDetection in EyeArea found by HaarCascades
         */
        protected function detectIris(r:Rectangle, detector:IrisDetector):Point
        {
            var area:Rectangle = r.clone();

            // Crop to fit eye area
            area.width *= 0.9;
            area.x += area.width * 0.0;
            area.height *= 0.7;
            area.y += area.height * 0.5;
            var clipRect:Rectangle = new Rectangle( area.x * scaleFactor, area.y * scaleFactor, area.width * scaleFactor, area.height * scaleFactor );
            var eyeScaleMatrix:Matrix = new Matrix( 1 * eyeScaleFactor, 0, 0, 1 * eyeScaleFactor );

            // Get a copy of the eye area from the camera input
            var eyeData:BitmapData = new BitmapData( clipRect.width, clipRect.height, false, 0 );
            eyeData.copyPixels( cam.bitmapData, clipRect, new Point() );

            // Upscale eye area
            var eyeDetectionMap:BitmapData = new BitmapData( clipRect.width * eyeScaleFactor, clipRect.height * eyeScaleFactor, false, 0 );
            // eyeData.clone();
            eyeDetectionMap.draw( eyeData, eyeScaleMatrix, null, "normal", null, false );

            // Create a temp. grayscale image of the eyearea to find the light intensity
            var tmpData:BitmapData = eyeDetectionMap.clone();
            tmpData.applyFilter( tmpData, tmpData.rect, new Point(), GRAYSCALE_MATRIX );
            // calculate average color value to detect if night mode should be used
            var histogram:Vector.<Vector.<Number>> = tmpData.histogram();
            var i:int;
            var red:Number = 0;
            var green:Number = 0;
            var blue:Number = 0;
            var countInverse:Number = 1 / (tmpData.width * tmpData.height);

            for(i = 0; i < 256; ++i)
            {
                red += i * histogram[0][i];
                green += i * histogram[1][i];
                blue += i * histogram[2][i];
            }
            red *= countInverse;
            green *= countInverse;
            blue *= countInverse;

            // var avarageColor:uint = 0xFF000000 | red << 16 | green << 8 | blue;
            var imageBrightness:int = (red << 16 & 0xff) + (green << 8 & 0xff) + blue & 0xff;

            // Adjust Image color settings depending on image average brightness
            if(imageBrightness < 80)
            {
                detector.brightness = 10;
                detector.contrast = 80;
                detector.saturation = 0;
                detector.hue = 0;
            }
            else if(imageBrightness >= 80 && imageBrightness < 127)
            {
                detector.brightness = 0;
                detector.contrast = 40;
                detector.saturation = 100;
                detector.hue = 0;
            }
```

Region um das Auge beschneiden

Kopie des Augenbildes aus dem Kamerabild erstellen

Hochskallieren

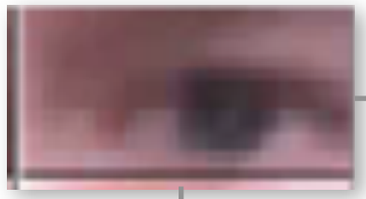
Helligkeit des Bereiches ermitteln,
um auf verschiedene Lichtverhältnisse zu reagieren
(Wertebereich: 0 - 255)

Entsprechend der Helligkeit,
die Farbwerte des Augenbildes neu justieren

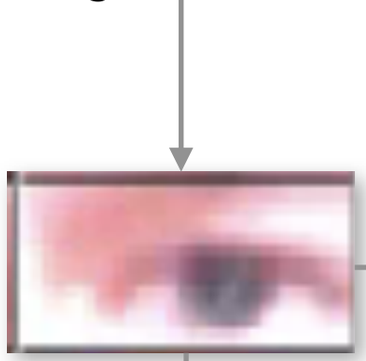
Klassenpfad: com.tgatzky.Interface.as

Realisierung

Iris-Erkennung



Automatische Nachbelichtung mit Ebeneneffekt: Hinzufügen



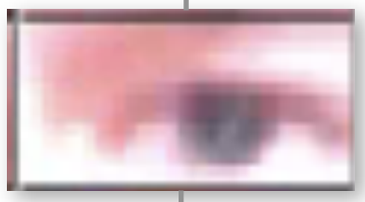
```
Interface.as   IrisDetector.as

97      /**
98       * Iris detection cylce
99       */
100     public function detect(bitmapData:BitmapData = null):BitmapData
101     {
102         if(bitmapData != null) source = bitmapData;
103         else source = this.source;
104
105         this.success = false;
106         foundPoints = new Vector.<Point>;
107
108         // Prepare bitmapData
109         var input:BitmapData = new BitmapData(source.width, source.height, false, 0);
110         input.lock();
111         input.draw(this.source, drawMatrix, null, "normal", null, smoothing);
112
113         var w:int = input.width;
114         var h:int = input.height;
115
116         // Reset ColorMatrix
117         colorMatrix.reset();
118
119         output = new BitmapData(w, h, false, 0);
120         var detectionMap:BitmapData =  new BitmapData(w,h,false,0);
121         var foreground:BitmapData = new BitmapData(w,h,false,0);
122         var background:BitmapData = new BitmapData(w,h,false,0);
123
124         detectionMap.lock();
125         foreground.lock();
126         background.lock();
127
128         foreground = input.clone();
129         background = input.clone();
130
131
132         // Enhance saturation and brightness via BlendMode ADD
133         foreground.draw(foreground, null, null, BlendMode.ADD);
134
135         // Sharp image using a high pass filter and BlendMode OVERLAY
136         //foreground.applyFilter(foreground, foreground.rect, new Point(), HIGHPASS)
137         //foreground.draw(foreground, null, null, BlendMode.OVERLAY);
138         detectionMap.draw(foreground);
139
140         // create copy for output
141         blendModeImage = detectionMap.clone();
142
143
144         // calc the average brightness of the image after enhancement
145         //imageBrightness = findAverageBrightness(blendModeImage);
146
147         // Adjust Image color settings
148         colorMatrix.adjustColor(brightness, contrast, saturation, hue);
149
150         // Apply ColorMatrix
151         var renderBmp:Bitmap = new Bitmap();
152         renderBmp.bitmapData = detectionMap.clone();
153         renderBmp.bitmapData.lock();
154         renderBmp.filters = [new ColorMatrixFilter(colorMatrix)];
155         renderBmp.bitmapData.draw(renderBmp);
156         detectionMap = renderBmp.bitmapData.clone();
157
158         renderImage = detectionM
159
```

Erstelle 2 identische Kopien des Ausgangsbildes

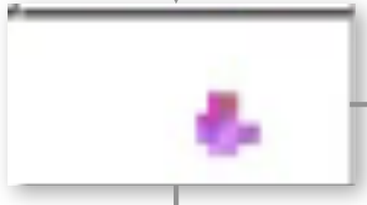
Setze den Ebeneneffet des oberen Bildes aus *Hinzufügen*

Klassenpfad: com.tgatzky.tracking.IrisDetector.as



Bildwerte setzen

Die häufigste Farbe finden



Häufigste Farbe mit Weiß ersetzen

```
// Enhance saturation and brightness via BlendMode ADD
foreground.draw(foreground, null, null, BlendMode.ADD);

// Sharp image using a high pass filter and BlendMode OVERLAY
```

```
/**
 * Finds the brightest color in a given bitmap
 * @param bitmapData BitmapData
 * @return uint
 */
protected function findBrightestColor(bitmapData:BitmapData):uint
{
    var w:int = bitmapData.width;
    var h:int = bitmapData.height;
    var tmp:BitmapData = bitmapData.clone();
    var color:uint;
    var data:Vector.<uint> = tmp.getVector(tmp.rect);

    var Rmax:int = 0;
    var Gmax:int = 0;
    var Bmax:int = 0;
    var Rcount:int = 0;
    var Gcount:int = 0;
    var Bcount:int = 0;

    for (var y:int = 0; y < h; ++y)
    {
        for (var x:int = 0; x < w; ++x)
        {
            if(data[x + (y * w )] == 0) continue;

            var R:int = ( data[x + (y * w )] >> 16 ) & 0xFF;
            var G:int = ( data[x + (y * w )] >> 8 ) & 0xFF;
            var B:int = ( data[x + (y * w )] ) & 0xFF;

            if(R > Rmax) Rmax = R;
            if(G > Gmax) Gmax = G;
            if(B > Bmax) Bmax = B;

            if(R > G && R > B) Rcount++;
            if(G > R && G > B) Gcount++;
            if(B > R && B > G) Bcount++;

        }
    }

    if(Rcount > Gcount && Rcount > Bcount)
        color = 0xFF000000 | (Rmax << 16 );
    if(Gcount > Rcount && Gcount > Bcount)
        color = 0xFF000000 | (Gmax << 8);
    if(Bcount > Rcount && Bcount > Gcount)
        color = 0xFF000000 | (Bmax);

    //trace("Rcount: " + Rcount + " Gcount: " + Gcount + " Bcount: " + Bcount);
    //trace("Rmax: " + Rmax + " Gmax: " + Gmax + " Bmax: " + Bmax);

    return color;
}
```

```
detectionMap.threshold(detectionMap, detectionMap.rect, new Point(), operation, thres, 0xffffffff);

_thresholdColor = thres;
```

Klassenpfad: com.tgatzky.tracking.IrisDetector.as

Falls das Auge geschlossen ist, wird die gesamte Hautfarbe mit Weiß überschrieben.

1. Binärbild-Erstellen (Schwarz/Weiß)
2. Alle Farben ausser Weiß auf Schwarz setzen
3. Weichzeichnung
4. Farbwertsuche

```
detectionMap.threshold(detectionMap, detectionMap.rect, new Point(), operation, thres, 0xFFFFFFFF);

_thresholdColor = thres;

// create copy for output
thresholdImage = detectionMap.clone();

// Grayscale
detectionMap.applyFilter(detectionMap, detectionMap.rect, new Point(), GRAYSCALE_MATRIX);

// Erase pupile reflextion by thresholding everything that is not white to black
detectionMap.threshold(detectionMap, detectionMap.rect, new Point(), "!=", 0xFFFFFFFF, 0xFF000000);

// Apply Blur for noise reduction
BLUR_FILTER.blurX = BLUR_FILTER.blurY = this.blurSize;
detectionMap.applyFilter(detectionMap, detectionMap.rect, new Point(), BLUR_FILTER);

// Soft out the iris to get a smooth blob
var t:int = 127;
var blobThres:uint = 0xFF000000 | (t << 16 | t << 8 | t);
detectionMap.threshold(detectionMap, detectionMap.rect, new Point(), ">", blobThres, 0xFFFFFFFF);

// Find iris by simple blob detection
var irisRegion:Rectangle = new Rectangle();
irisRegion = detectionMap.getColorBoundsRect(0xFFFFFFFF, 0xFF000000, true);
this.irisRectangle = irisRegion;

blobImage = detectionMap.clone();
//trace("w: " + irisRegion.width + "  " + this.minIrisWidth)
//trace("h: " + irisRegion.height + "  " + this.minIrisHeight)

var avarage:Number;
var tmp:int = 0;
if( !irisRegion.isEmpty() || irisRegion.width != 0 || irisRegion.height != 0 )
{
    if(irisRegion.width < this.minIrisRadius*2 || irisRegion.height < this.minIrisRadius*2)
    {
        output = detectionMap.clone();
        return output;
    }

    // Create a copy of the iris image from the input bitmap data
    this.iris = new BitmapData(irisRegion.width, irisRegion.height, false, 0);
    iris.copyPixels(source, irisRegion, new Point());

    // calculate avarage radius for hough transformation
    var rw:Number = irisRegion.width;
    var rh:Number = irisRegion.height;
    var currRadius:Number = (rw + rh) / 2 ;
    var elem:int;

    // try to flat out the iris radius by calculating the avarage of the last radi found
    if(irisRadiusVec.length < maxRadiToCompare)
    {
        irisRadiusVec.push(currRadius);
        for each (elem in irisRadiusVec)
        {
            tmp += elem
        }
```

Klassenpfad: com.tgatzky.tracking.IrisDetector.as

Realisierung

Bewegungserkennung und Feedbacks

Beispiel: *Swipe/Wisch*-Bewegung und EventHandling

```
/**
 * Called when the motion ended. Calculates the swipe motion properties and dispatches Events
 */
protected function onMotionEnd(e:MotionTrackerEvent):void
{
    // relative distance between start and end position
    var dx:Number = endX - startX;
    var dy:Number = endY - startY;

    swipeLength = Math.round( FastMath.sqrt( (dx*dx) + (dy*dy) ) );

    if(swipeLength >= minSwipeDistance)
    {
        swipeAngle = calculateAngle( dy, dy );
        if(dx < 0 && !isSwipping)
        {
            dispatchEvent( new MotionTrackerEvent( MotionTrackerEvent.MOTION_SWIPE_RIGHT_TO_LEFT ) );
        }
        else if(dx > 0 && !isSwipping)
        {
            dispatchEvent( new MotionTrackerEvent( MotionTrackerEvent.MOTION_SWIPE_LEFT_TO_RIGHT ) );
        }
        else if( dy > 0 && !isSwipping )
        {
            dispatchEvent( new MotionTrackerEvent( MotionTrackerEver
        }
        else if( dy < 0 && !isSwipping )
        {
            dispatchEvent( new MotionTrackerEvent( MotionTrackerEver
        }
        else
        {

        }
    }
    resetMotionVariables();

}
```

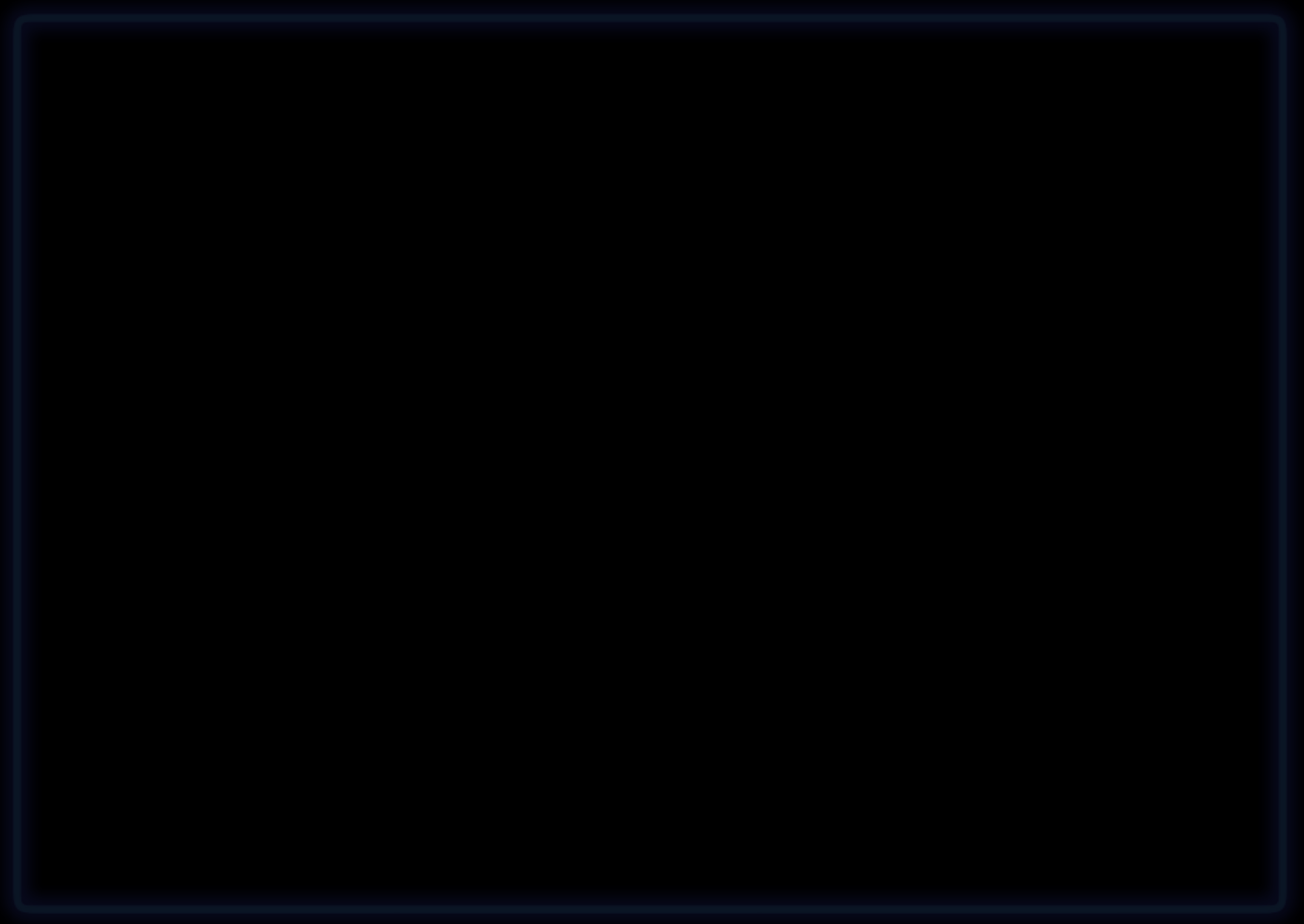
Aktion **Reaktion**

```
/**
 * MotionTracker Eventhandlers
 */
private function onMotionSwipeLeftToRight(e:MotionTrackerEvent = null):void
{
    monoslideshow.previousImage();
}
private function onMotionSwipeRightToLeft(e:MotionTrackerEvent = null):void
{
    monoslideshow.nextImage();
}
```

Klassenpfad: com.tgatzky.motion.MotionTracker.as

Beispiel

Eine selbstscrollende Internetseite, die sich der Lesegeschwindigkeit des Benutzers anpasst und auf seine Körpergesten entsprechend reagiert.



Ausblick

- Handerkennung ohne Bewegungserkennung
- Mehr Gesten und eigene Gesten, zeichnerisch
- Event. Gesichtserkennung einbauen
- Event. Akustische Feedbacks einbauen
- Erstes reales Projekt starten

Vielen Dank für Ihre Aufmerksamkeit

Betreuer

Prof. Hartmut Ginnow-Merkert

Prof. Dr. Heik Afheldt