Installation

- Download this (REVU) repository to your disk (anywhere)
- Duplicate the folder "ImageTargets-2-0-6" in VuforiaSDK/Samples
- rename to what you want (e.g. AiTargets)
- open with finder the folder
- copy the folder "REVU_IOS" (and ist contents too) to your new project folder (same location as "media", "ArCommon"…)
- open in that folder "ImageTargets.xcodeproj" with Xcode
- Drag the folder "REVU_IOS" from the Finder (now located in your project-folder) into Xcode's project-explorer (this is the left side) - be sure you have check-marks for "*create groups for any added folder*" and "*Add to targets"*

Now you have to make some modifications in the sample-code:

in "EAGLView.h"

add

```
#import "REVuModel.h"
```

below

```
#import "AR_EAGLView.h"
```

add

```
REVuModel * model1;
REVuModel * model2;
REVuModel * model3;
```

between { ... and .... }

in „EAGLView.mm"

replace `- (void) setup3dObjects { ...}` with this:

```
- (void) setup3dObjects
{
    NSLog(@"############## LOADING MODELS ###################################");
   //THIS COULD TAKE SOME TIME ... SO BE PATIENT !

    //LOADED MODELS ARE PRE-SCALED SO THAT THEY FIT INTO OUR VIEWPORT
    NSString* dataPath = [[NSBundle mainBundle] resourcePath] ; // WITHOUT TRAILING SLASH !!

    NSString * m1 = @"Apple.3ds";
    model1 = new REVuModel([dataPath UTF8String], [m1 UTF8String]) ;
    if(model1->isValid)
    {
        NSLog(@"Model %@ loaded !",m1);
        model1->TranslateTo(0, 0, 0);
        model1->ScaleUniformTo(0.4);            //ITS TOOOO BIG ... SO SCALE DOWN
        model1->RotateTo(0, 0, 0);
        model1->Alpha(1.0);
        model1->SwitchLight(true);

    }
```

```
    NSString * m2 = @"astroBoy.dae";
    model2 = new REVuModel([dataPath UTF8String], [m2 UTF8String]) ;
    if(model2->isValid)
    {
        NSLog(@"Model %@ loaded !", m2);
        model2->ScaleUniformTo(0.6);           // A LIITLE BIT SMALLER
        model2->RotateTo(90.0, 0.0, 0.0);
        model2->TranslateTo(40, 40, 0);              // A LITTLE BIT ABOVE THE TARGET
        model2->LightDirection(0.0, 0.0, -1.0);
        model2->SwitchLight(false);

    }

    NSString * m3 = @"Torus.obj";
    model3 = new REVuModel([dataPath UTF8String], [m3 UTF8String]) ;
    if(model3->isValid)
    {
        NSLog(@"Model %@ loaded !", m3);
        model3->ScaleUniformTo(0.4);           // A LIITLE BIT SMALLER
        model3->RotateTo(0.0, 0.0, 0.0);
        model3->TranslateTo(0, 0, 0);               // A LITTLE BIT ABOVE THE TARGET
        model3->LightDirection(0.0, 0.0, -1.0);
        model3->SwitchLight(true);

     }

    NSLog(@"############# FINISH LOADING MODELS ###################################");

}
```

replace `- (void)renderFrameQCAR{ ...}`
with this:

```
- (void)renderFrameQCAR
{
    [self setFramebuffer];

    glClearColor(0.0f, 0.0f, 0.0f,0.0f);

    // Clear colour and depth buffers
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    // Render video background and retrieve tracking state
    QCAR::State state = QCAR::Renderer::getInstance().begin();
    QCAR::Renderer::getInstance().drawVideoBackground();

    //NSLog(@"active trackables: %d", state.getNumActiveTrackables());

    // We must detect if background reflection is active and adjust the culling direction.
    // If the reflection is active, this means the pose matrix has been reflected as well,
    // therefore standard counter clockwise face culling will result in "inside out" models.
    if(QCAR::Renderer::getInstance().getVideoBackgroundConfig().mReflection ==
QCAR::VIDEO_BACKGROUND_REFLECTION_ON)
        glFrontFace(GL_CW);  //Front camera
    else
        glFrontFace(GL_CCW);   //Back camera

    for (int i = 0; i < state.getNumTrackableResults(); ++i)
    {
        // Get the trackable
        const QCAR::TrackableResult* result = state.getTrackableResult(i);
```

```
        const QCAR::Trackable& trackable = result->getTrackable();
        QCAR::Matrix44F modelViewMatrix = QCAR::Tool::convertPose2GLMatrix(result-
>getPose());


        // Choose the model based on the target name
        if (!strcmp(trackable.getName(), "chips"))
        {
            if(model1)
            {
                if(model1->isValid)
                {
                    model1->RotateBy(0.0, 0.0, 1.0); //ROTATE  ....
                    model1->Draw(modelViewMatrix, qUtils.projectionMatrix);
                }
            }
        }


        if (!strcmp(trackable.getName(), "stones"))
        {

            if(model3)
            {
                if(model3->isValid)
                {

                    model3->RotateBy(1.0, 1.0, 1.0); //ROTATE  ....
                    model3->Draw(modelViewMatrix, qUtils.projectionMatrix);
                }
            }

            if(model2)
            {
                if(model2->isValid)
                {
                    model2->Draw(modelViewMatrix, qUtils.projectionMatrix);
                }
            }

        }

    }

    QCAR::Renderer::getInstance().end();
    [self presentFramebuffer];
}
```

Run this on your device with Targets „Chips“ and „Stones“ ...

Important:

If your model(s) uses PNG’s make sure you have to make this adjustment in Xcode:

Go to: Build Settings -> Packaging and set „Compress PNG Files“ to „NO“ !!!

After loading your model(s) you can modify some properties:

Rotation, Translation and Scaling:

„absolute“

```
model->RotateTo(x, y, z);
model->TranslateTo(x, y, z);
model->ScaleTo(x, y, z);
model->ScaleUniformTo(scale);
```

„relative“

```
model->RotateBy(x, y, z);
model->TranslateBy(x, y, z);
model->ScaleBy(x, y, z);
model->ScaleUniformBy(scale);
```

To set the „transparency“ (Alpha) - even with textures

```
model->Alpha(x);   (0.0 < x < 1.0)
```

To assign a new material (overrrides models material)

```
Model->SetMaterial(Material(GOLD));   // see material.h for pre-defined materials
```

To assign a new texture (overrides models texture)

```
NSString * t1 = @"whateveryouwant.jpg";
model1->SetTexture([dataPath UTF8String], [t1 UTF8String]);
```

Have Fun !