Interpolating points between drag events using Symbian C++

Overview

Interpolation is a method of constructing new data points within the range of a discrete set of known data points.

When dragging an item on the screen and that item is updated after receiving a `TPointerEvent::EDrag` event, it moves remarkably. The reason is that the dragging points received by the application are quite far from each other. The following interpolation calculates the missing points between two `TPointerEvent::EDrag` events and draws the item also in them.

NOTE: The application has an Interpolation menu item which you can use to enable/disable interpolation and see the difference. Interpolation is disabled by default.

This snippet can be self-signed.

MMP file

The following capabilities and libraries are required:

```
CAPABILITY None

LIBRARY avkon.lib
```

Header file

```
#include <coecntrl.h>
class CExampleStubAppView : public CCoeControl
    {
    ...
    public:
        void EnableInterpolation(TBool aEnable);

    private:
        void StartDrawingL();
        void StopDrawing();
        static TInt DoDraw(TAny* aObj);
        void HandlePointerEventL(const TPointerEvent&
        aPointerEvent);

        // Interpolation is calculated here
        void Move(const TPoint& aPoint);

    private:
        CPeriodic*          iPeriodic;
        TRect iPictureRect;
        TPoint iPicturePoint;
        TPoint iStylusPoint;
        TBool iDragging;
        TBool iInterpolate;
    };
```


(C) Copyright Nokia 2014. All rights reserved.

Source file (consts for interpolation)

```
const TReal32 KSPEED = 8;           // Moving speed
const TInt DELAY = 30000;           // = 33 fps (Frame/Seconds)
const TReal KFRAMETIME = 0.03030;   // 1/33 fps
```

Source file (general)

```
void CExampleStubAppView::ConstructL( const TRect& aRect )
    {
    ...
    ActivateL();

    // Start updating screen in 33 fps speed
    StartDrawingL();
    }

void CExampleStubAppView::Draw( const TRect& /*aRect*/ ) const
    {
    CWindowGc& gc = SystemGc();
    TRect drawRect( Rect());
    gc.Clear( drawRect );

    if (iDragging)
        {
        gc.SetBrushColor(KRgbRed);
        gc.SetBrushStyle(CGraphicsContext::ESolidBrush);
        gc.DrawRect(iPictureRect);
        }
    }

void CExampleStubAppView::StartDrawingL()
    {
    if (!iPeriodic)
        {
        iPeriodic = CPeriodic::NewL(CActive::EPriorityIdle);
        }
    iPeriodic->Start(TTimeIntervalMicroSeconds32(0),
       DELAY, TCallBack(CExampleStubAppView::DoDraw, this));
    }

void CExampleStubAppView::StopDrawing()
    {
    iPeriodic->Cancel();
    }

TInt CExampleStubAppView::DoDraw(TAny* aObj)
    {
    static_cast<CExampleStubAppView*>(aObj)->DrawNow();
    return 1;
    }
```


(C) Copyright Nokia 2014. All rights reserved.

```
void CExampleStubAppView::HandlePointerEventL(const TPointerEvent&
aPointerEvent)
    {
    if (aPointerEvent.iType == TPointerEvent::EButton1Down)
        {
        iPicturePoint = aPointerEvent.iPosition;
        iPictureRect.SetRect(iPicturePoint,TSize(10,10));
        iDragging = ETrue;
        }
    else if(aPointerEvent.iType == TPointerEvent::EDrag)
        {
        Move(aPointerEvent.iPosition);
        }
    else if(aPointerEvent.iType == TPointerEvent::EButton1Up)
        {
        iDragging = EFalse;
        }
    }

void CExampleStubAppView::EnableInterpolation(TBool aEnable)
    {
    iInterpolate = aEnable;
    }
```

Source file (actual interpolation logic)

```
void CExampleStubAppView::Move(const TPoint& aPoint)
    {

    // No interpolation
    if (!iInterpolate)
        {
        TPoint offset;
        offset.iX =  aPoint.iX - iPictureRect.iTl.iX;
        offset.iY = aPoint.iY - iPictureRect.iTl.iY;
        iPictureRect.Move(offset);
        return;
        }

    // Use interpolation

    // Stylus is here
    iStylusPoint = aPoint;

    StopDrawing();
    TBool move = EFalse;

    // Calculate picture smooth drawing to stylus position (interpolate)
    if (iPicturePoint.iY < iStylusPoint.iY)
        {
        iPicturePoint.iY += ((iStylusPoint.iY - iPicturePoint.iY) * KSPEED) *
KFRAMETIME;
        move = ETrue;
```


(C) Copyright Nokia 2014. All rights reserved.

```
        }
    if (iPicturePoint.iY > iStylusPoint.iY)
        {
        iPicturePoint.iY -= ((iPicturePoint.iY - iStylusPoint.iY) * KSPEED) *
KFRAMETIME;
        move = ETrue;
        }
    if (iPicturePoint.iX < iStylusPoint.iX)
        {
        iPicturePoint.iX  += ((iStylusPoint.iX - iPicturePoint.iX ) * KSPEED) *
KFRAMETIME;
        move = ETrue;
        }
    if (iPicturePoint.iX  > iStylusPoint.iX)
        {
        iPicturePoint.iX  -= ((iPicturePoint.iX  - iStylusPoint.iX) * KSPEED) *
KFRAMETIME;
        move = ETrue;
        }

    if (move)
        {
        iPictureRect = TRect(iPicturePoint,TSize(10,10));
        StartDrawingL();
        }
    }
```

Postconditions

The object is moving smoothly on the screen when dragging it with the stylus.

See also

Full example source code: File:Interpolation ExampleStub.zip

Interpolation in Wikipedia.org

(C) Copyright Nokia 2014. All rights reserved.