Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2014) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

6th International conference on Intelligent Human Computer Interaction, IHCI 2014

Shapers: capturing free form shapes for bendable device interactions

Samudrala Nagaraju

Web & Services Team, Samsung R&D Institute India, Bangalore, India.

Abstract

Future mobile and wearable devices are heavily influenced by advances in materials which make the devices flexible to interact. In this paper, we propose a system to capture the shape of bent device at graphics framework and make use of the shape to develop novel user interactions for native and app store downloadable applications on consumer electronic devices. We further detail the API, parameters and prototype on Android for shape generation along with OpenGL based emulation for bendable device deformation, followed by usability study results.

Peer-review under responsibility of the Scientific Committee of IHCI 2014.

Keywords: Bendable displays; Flexible displays; Graphics Framework; Bezier curves; NURBS; Tangible interaction; Gestures

1. Introduction

Technological background of this work is based on free form deformable devices (Fig.1(i)), possible because of materials - wrinkle-free single-crystal monolayer graphene on a silicon wafer [1], and one atom thick graphene device structure (inexpensive to mass produce from **Samsung-SKKU joint research)** [2]. Research in bendable devices is focused on interaction styles [3], automatic shape change [4] and folding gestures [5].

2. Shape Representation & Proposed System

Use cases possible from device deformation shape capture in software depends on the flexibility of shape representation in mathematical form. Considering this, shape is represented as curves and surfaces as shown in Fig.1 (ii). Second, shape size and position independent gestures are researched by Kristen Warren et al. [5] and conclude that users do not remember the size and position to recreate a gesture. Considering this recommendation, the shape representation needs to be size independent, position independent and also invariant under affine transform.

Curves: When a device is bent, each position represents a different curve. For example, Fig 1(ii)(A) and 1(ii)(B) are uniform curves and Fig 1(ii)(C) is non-uniform curve in vertical direction for left, middle and right positions.

Peer-review under responsibility of the Scientific Committee of IHCI 2014.

Surface representation is in two forms, 3D surface and sweep surface. **Sweep surfaces** are constructed from curves [6] as shown in Fig 1(ii)(c). B1, B2 and B3 are the curves generated for device shape. A 2-rail sweep surface uses at least three curves, defining the two edges of the surface and the other defining the surface's cross sections [6].

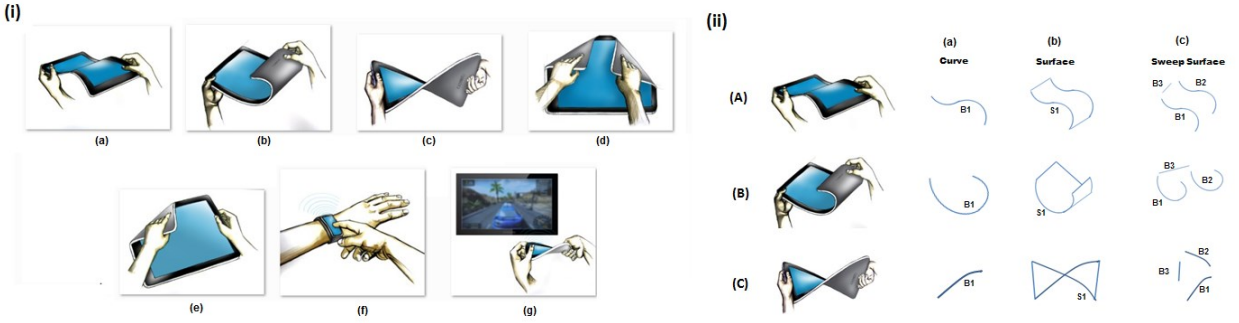

Fig.1 **(i) Interaction with bendable devices**. Single hand bend: (b), (e), (f); Both hands bend: (a), (c), (d), (g); Single device interaction: (a) to (f); Multi device interaction: (g); Tablet device: (a), (b), (c), (d), (e); Wearable device: (f); Mobile device: (g); **(ii) Shape Representation** (a) Curve – Left position of device (b) Surface (c) Sweep surface

B-splines (non-uniform non-rational B-splines) and **NURBS** (Rational B-splines) forms are used for shape representation as they consume less memory and easy to perform operations [7]. NURBS is generalization of rational Bezier, which in turn is generalization of Bezier. An application uses one of the Bezier forms based on its shape usage. For example, a conic curve can be represented using a rational function and cannot be represented using a Bezier curve and B-splines are used for the approximation of a circle where NURBS cannot be used [8]. Based on research work done by one team in our organization on mobile devices, sweep surfaces take 6 to 8 times less processing time when compared to 3D surfaces, which make them ideal to represent complex graphics features on mobile devices to improve FPS, consume less power, consume less memory and perform complex operations.

(i)

(a)

New Interface	Description
android.view.View.OnBendShapeListener	Interface definition for a callback to be invoked when a be nd shape event is dispatched to this view
abstract boolean onBendShape (View v, ShapeEvent event)	Called when a Shapers event is dispatched to a view.

(b)

View Class Extension	Description
public void setOnShapeBendListener (View.OnBendShapeListener l, int shape, int shapeform ShapePosition sp, ShapeTransform transform, SpeedParams spdparam, Region r)	Register a callback to be invoked when the mobile is bent

(c)

New Class - ShapeEvent	Description
java.lang.Object android.view.InputEvent android.view.ShapeEvent	Object used to report shape events along with speed para meters
Methods	
final int getShape()	SHAPE_CURVE SHAPE_SWEEP_SURFACE
final int getShapeForm()	SHAPE_FORM_BSPLINE SHAPE_FORM_NURBS
final Point[ ][ ] getShapeParams()	Returns the parameters of bend shape event generated
final ShapeTransform getTransformParams()	Returns the transform parameters of shape

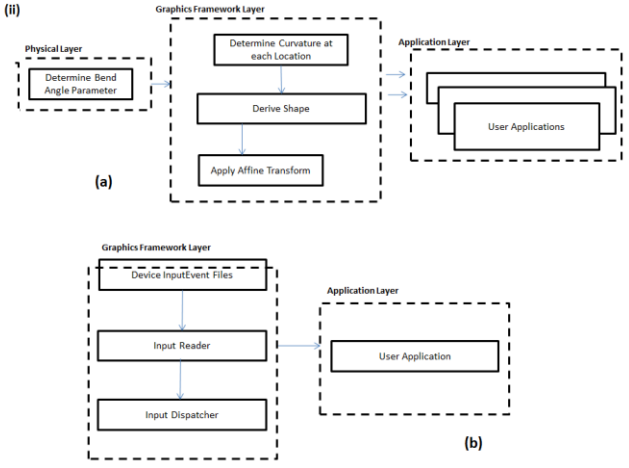

Fig. 2. **Shapers Interface** (i) Android platform API (ii) Shape generation flow chart (a) General (b) Android platform

Shape Generation: Kristen Warren et al.[5] identifies parameters influencing bend gesture classification and Anne Roudant et al. [4] identifies factors for bending actuated flexible devices. Shape capture parameters are divided into 3 categories – Shape generation (area, granularity) parameters are influenced from [4], Shape change parameters (speed, duration) are influenced from [5] and application usage (Aspect ratio – A region in application layout where the shape is consumed e.g. image effects, Transformations – affine transformations required for application usage) parameters are devised as "Shapers" needs to cater both native and downloadable apps. Shape

generation requires multiple bend sensor inputs to derive the shape (curve, surface) at graphics framework and apply affine transformation before passing to application as shown in Fig 2(i), and Fig 2(ii) is flow on Android platform.

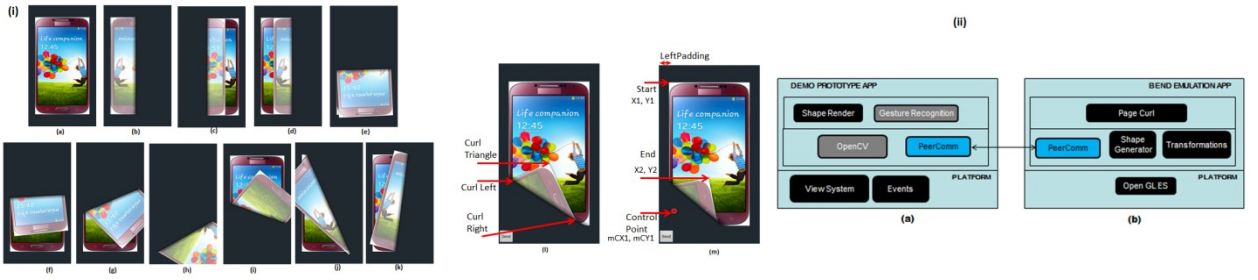

Fig. 3. **(i) Shape Generation** (a) Original position (b) to (d) 1-way bends (e) to (k) 2-way bends (l) Bend emulation using page curl (m) Control point calculation for curve generation. **(ii) Demo Setup with two mobile phones** (a) Target app (b) Bend emulator app

"Shapers" Design: Application registers for shape capture with device API. Shape is generated based on bend sensor inputs and sent to application via graphics framework. Shape**:** Based on reasoning given above, curve and sweep surface shapes, and rational b-spline and NURBS mathematical forms are used. Shape Position**:** In case of curve, alignment direction – horizontal or vertical - along with position (left, middle, right) parameter are requested. Affine Transforms**:** A function between affine spaces preserving points, straight lines and planes is applied post shape generation e.g. scaling (*2, *0.5, etc.), rotation, reflection, shearing. Speed**:** A shape is arrived on change in movement of device from an initial position to shape position at various speeds (Speed and duration parameters) is not detailed in this paper. Target Region**:** This is a rectangular area in a widget or layout in application. An additional step is computed for uniform or non-uniform scaling of shape, if set. Not required in gesture recognition cases. "Shapers" is realized for Android platform as in Fig 2(i). View class is extended and new interfaces are introduced. ShapeEvent is a new class to capture bend shapes and a new method to register for bend shape is added to View class. As graphics framework receives bend shape event, callback of OnBendShapeListerner is trigged.

3. Bend Emulation & Prototype Setup

Shape event generation requires construction of a hardware prototype which is limited with stiffness and deformation range. In this paper, we explore software and usability aspects, and so we chose to do bend emulation as an android app on mobile. **Page Curl**, a widely known graphics effect, is used for bend emulation [9]. Android page curl [10] is OpenGL based research friendly open source is used for bend emulation (Samsung galaxy S5) as in Fig. 3(i). While turning the page by finger, curl effect vertices are determined (Fig 3(i)(l)), which are used for control point calculation (Fig 3(i) (m)). **mCX1** = X1- LeftPadding/2; **mCX2** = 2 * (Y2- TopPadding-BottomPadding) – (Y2/2 - X2/2); Shape generation procedure discussed till here is extendable to sweep surfaces, as it needs generation of additional curves B1, B2 and B3 in a similar manner (Fig.1 (ii) c).

On demo prototype mobile, changes in view system are done for event generation and gesture application is developed (Fig. 3(ii)). Setup with 2 mobile phones over Wi-Fi usage is: (1) Launch peer connection app on 2 mobiles. (2) Connect over IP address (3) Launch bend emulation app and gesture applications. (4) Perform device bend on bend emulation app and send shape to demo prototype (5) Gesture app handles the shape event using an OpenCV based open source [11]. Fig 4(i) contains shapes match and not match gestures.

4. Usability Study

Bend emulation and three paper based low fidelity prototypes are given to users. Three user sessions were conducted with people from software, user experience and novice backgrounds (Age group: 24 and 35, Total: 9). **User Sessions:** Two sessions are formal for an exploratory study and third session is an informal discussion. Formal sessions consist of three phases. First phase consist of prototype explanation, paper based interactions, "Shapers" introduction based on Fig 1(i) and Fig 1(ii). Second phase is about target application and bend emulation usage

based on three scenarios as in Fig 4(i). Third phase is about paper based prototype interaction and asking questions (1) how do you feel the usage of "Shapers" for gestures? (2) What are the interesting cases in your daily life? (3) How will you hold the device for making shapes? (4) Make 5 shapes using paper prototype. Informal discussion session had first and third phases. This session is to capture the usability parameters and future direction without influence from prototype. **User Feedback**: (1) Users were comfortable making bigger shapes on smaller devices and were always making smaller shapes on bigger devices. (2) Participants preferred shapes till 2 bends as in Fig 4(ii)(3)(d). (3) Participants felt that gesture recognition independent of bend shape size and position is easy to remember. (4) Complex shapes are usable for multi device shown in Fig 1(i)(g) (5). Some felt that "Shapers" is an alternative to accelerometer usage. Based on user feedback, "Shapers" usage parameters are shown in Fig.4 (ii).

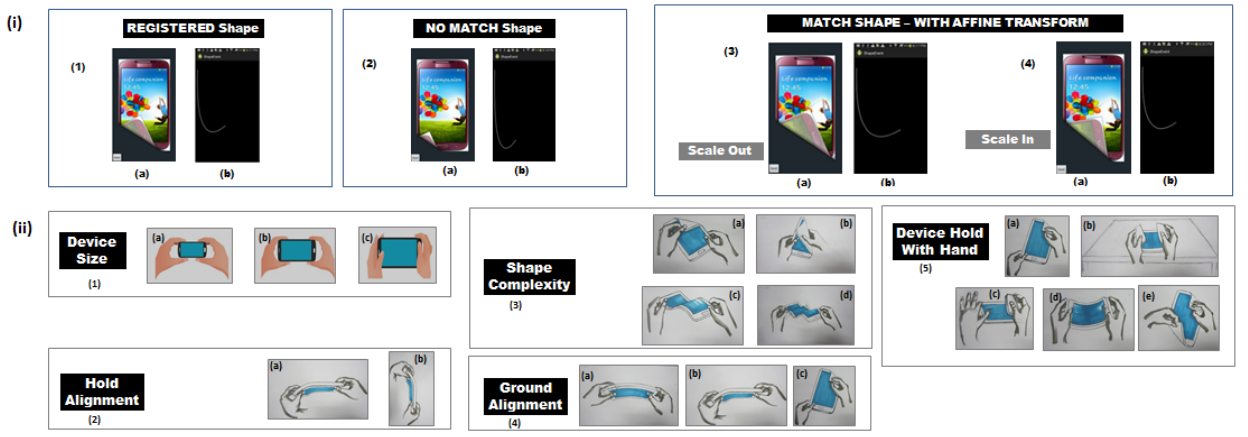

Fig. 4. **(i) Gesture Recognition Application** Scenarios (1) Gesture registered shape (2) Gesture NO match (3) Gesture match with an affine transform – Scale Out (4) Gesture match with an affine transform – Scale in. Each scenario contains a tuple (a) Bend emulator shape (b) Generated curve from bend emulated shape. **(ii) User Experience Parameters**. (1) Device size: (a) small (b) medium (c) large; (2) Hold alignment: (a) horizontal (b) vertical; (3) Shape complexity: (a) (b) single bend (c) double bend (d) three bend; (4) Ground alignment: (a) vertical (b) slanted (c) horizontal; (5) Device hold with hand (a) one hand edge (b) tabled (c) one hand side (d) two hand side (e) two hand edge.

5. Conclusion

In this paper, we introduce curve and sweep surface capture from bendable devices using B-Spline and NURBS representations, because of the flexibility they offer for affine transformation and compact memory storage. We also identify the design aspects and influential factors for "Shapers" API on android device platform and develop shape size independent gestures, and present usability parameters. We believe that "Shapers" would pave way for novel user interaction and features for bendable devices with fine local control in shape representation.

References

1. JH Lee, EK Lee, WJ Joo, Y Jang, BS Kim, JY Lim, SH Choi, SJ Ahn, JR Ahn, MH Park, CW Yang, BL Choi, SW Hwang, D Whang. Wafer-Scale Growth of Single-Crystal Monolayer Graphene on Reusable Hydrogen-Terminated Germanium. Science Express Journal, April 2014
2. Samsung Electronics and SKKU -http://spectrum.ieee.org/nanoclast/semiconductors/materials/singlecrystal-monolayer-graphene-produced-in-bulk-for-first-time. Retreived on 2014-4-11
3. Samudrala Nagaraju. Novel user interaction styles with flexible/rollable screens. CHItaly '13, Article No. 20
4. Anne Roudaut, Abhijit Karnik, Markus L., Sriram S. Morphees: toward high shape resolution in self-actuated flexible mobile devices. CHI'13
5. Kristen W., Jessica L., Vaibhav V. and Audrey Girouard. Bending the Rules: Bend Gesture Classification for Flexible Displays. CHI'13
6. Sweep surfaces : http://www.3dmax-tutorials.com/_2_Rail_Sweep_Surface.html. Retrieved on 2014-04-04
7. B-Spline: http://en.wikipedia.org/wiki/Non-uniform_rational_B-spline
8. LIBNURBS. http://libnurbs.sourceforge.net/old/nurbsintro.pdf
9. Flip Page Effect. http://en.wikipedia.org/wiki/Flip_page
10. Page Curl. https://github.com/harism/android_page_curl
11. Farzin Mokhtarian, Sadegh Abbasi. Pattern Recognition: Shape similarity retrieval under affine transforms. January 2002, Pages 31–41.