Flexible Live-Wire: Image Segmentation with Floating Anchors

Supplemental Material: User Feedback

B. Summa[1], N. Faraj[1], C. Licorish[1], and V. Pascucci[2]

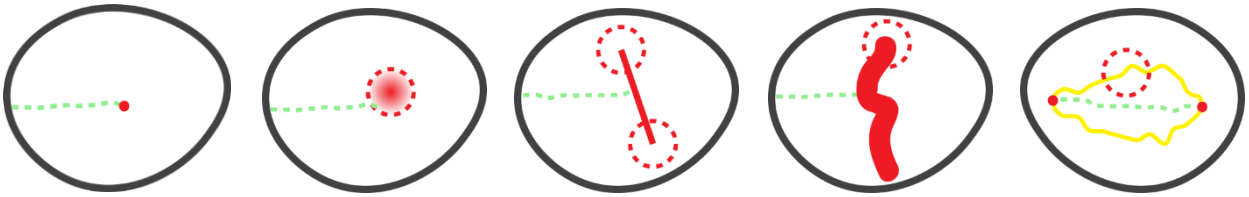
[1]Tulane University, United States
[2]SCI Institute, University of Utah, United States

User feedback

In this document, we describe the procedure used to elicit user feedback on our flexible anchors compared to the standard approach and the resulting survey. We performed the survey on both touch display and classical keyboard + mouse configurations. To represent the standard approach we used Adobe Photoshop's Magnetic lasso tool in the classical keyboard + mouse configuration. Due to the OS of our touch device, a node-anchor-only version of our software represented the standard.

Training

We explained to the user how to every anchor available in our system displayed bellow.



Node Probable Pinch Paint Pick

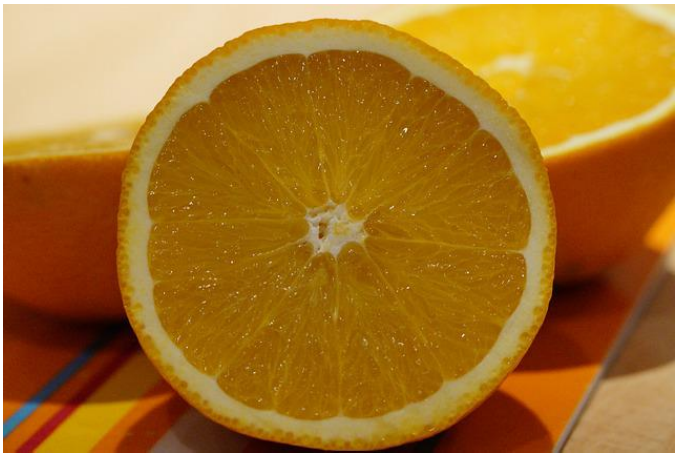
The user then train by segmenting the boat in the image bellow, when the user felt comfortable with the system she proceeded to perform the tasks described in the following section.

Tasks

Users were asked to perform the following tasks:

1. Segment the flower using the standard approach (node-only) and our new paint anchor:

2. Perform two segmentations of the orange, the full orange then the inner part i.e. without the peel using the standard approach (node-only) and our new flexible anchors:



3. Segment the pinwheel with their choice of anchors:

Once the users had performed the prescribed tasks, they were asked to fill the following survey to collect their impressions on our system.

Survey

Device Type

Touch Mouse+Keyboard

How often do you use image processing software?

1	2	3	4	5
Never				Always

Compared to the standard approach, how did you find the new approach?

1	2	3	4	5
Harder				Easier

Compared to the standard approach, did you arrive at your desired segmentation:

1	2	3	4	5
Slower				Faster

How did you find the flexibility of constraint choice?

1 2 3 4 5

Not Useful Useful

Please rank the following anchors from most (1) to least (5) useful?

_____ Node

_____ Paint

_____ Pinch

_____ Probable

_____ Pick

Would you use a paint, pinch, probable, or pick anchor if available in image processing software?

Yes No

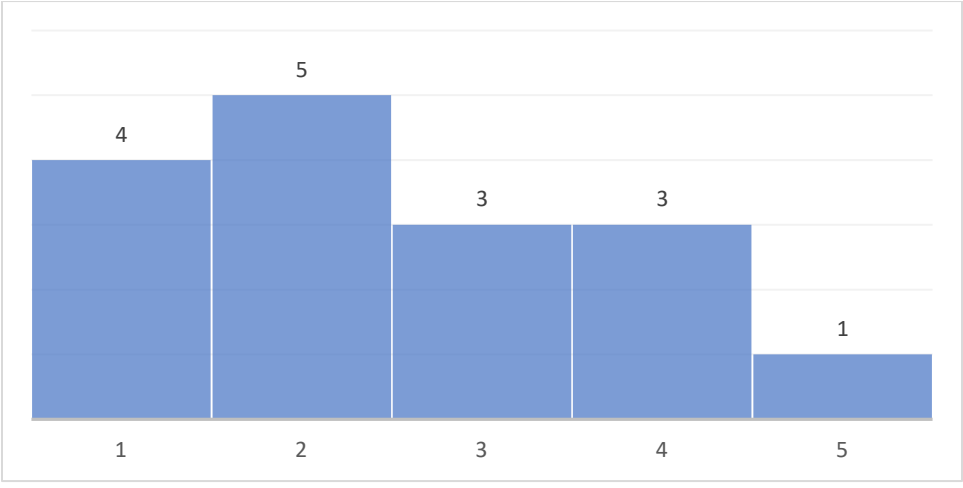
Results

In this section, we present the collected data from the above-mentioned survey.

Device Type

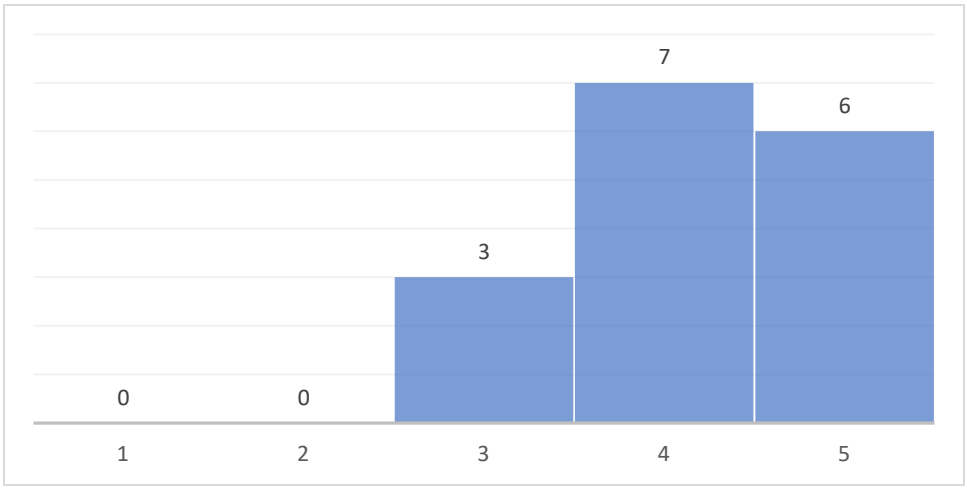
7 Touch 9 Mouse+Keyboard

How often do you use image processing software?



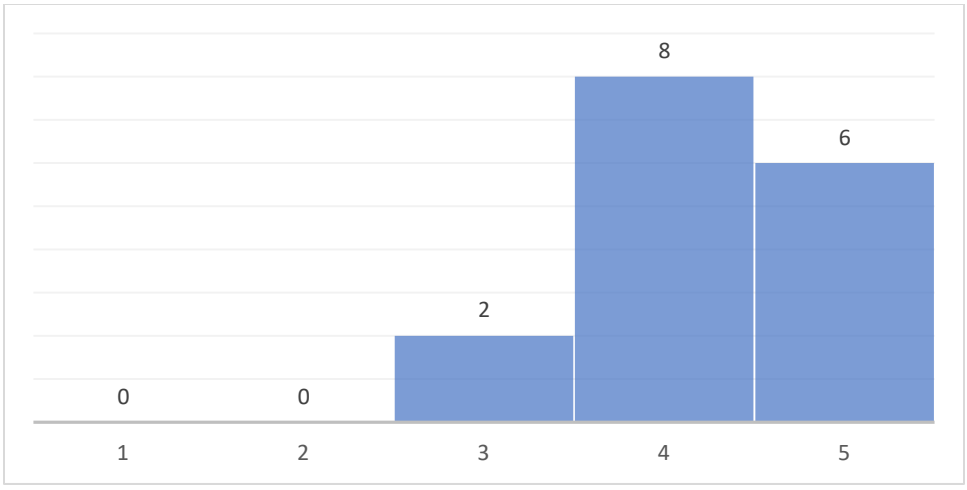
Never Always

Compared to the standard approach, how did you find the new approach?



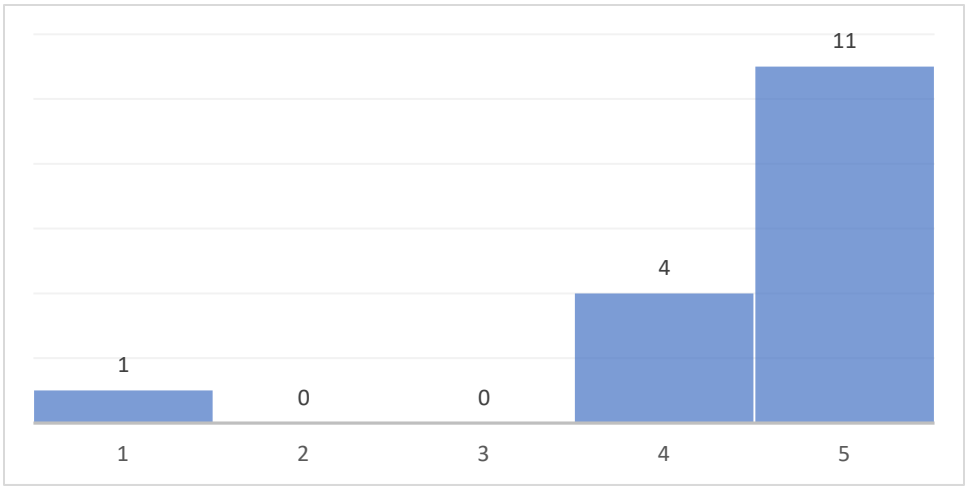
Harder Easier

Compared to the standard approach, did you arrive at your desired segmentation:



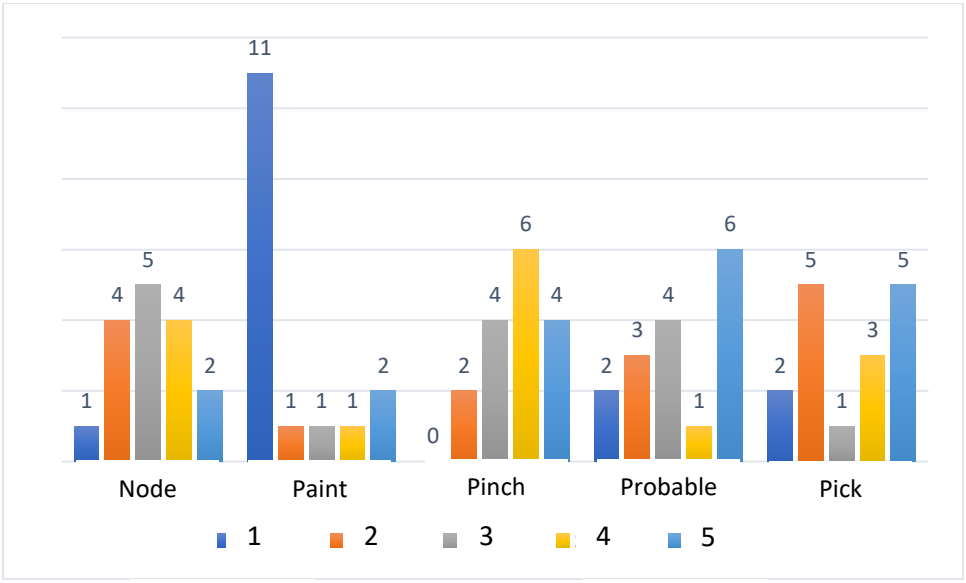
Slower Faster

How did you find the flexibility of constraint choice?



Not useful Useful

Please rank the following anchors from most (1) to least (5) useful?



Most useful Least useful

Would you use a paint, pinch, probable, or pick anchor if available in image processing software?

14	1
Yes	No

Timings in seconds for the 7 touch screen users

Flower standard live-wire	**Flower paint anchor**	**Orange standard live-wire**		**Orange ours both**
		outer+inner	**both**	
60	20	15+11	26	18
67	30	27+13	40	27
32	20	11+9	20	25
39	15	15+13	28	6
45	37	15+10	25	18
49	27	14+16	30	15
63	39	13+15	28	11

EUROGRAPHICS 2018 / D. Gutierrez and A. Sheffer
(Guest Editors)

Volume 37 (2018), Number 2

Supplemental: Flexible Live-Wire: Image Segmentation with Floating Anchors

B. Summa[1], N. Faraj[1], C. Licorish[1], and V. Pascucci[2]

[1]Tulane University, United States
[2]SCI Institute, University of Utah, United States

Algorithm 1 UpdateAnchorSet

Interactive update and minimum path lookup during anchor set adjustment:
Input: anchor set: A_i; set of new nodes for anchor: $\{n\}$
Output: minimum path: $\mathcal{P}$
$A_i \leftarrow setNodes(\{n\})$ ▷ update nodes
$m \leftarrow FindMinCostNode(A_i)$ ▷ $cost = cost(\mathcal{T}_i^{\mathcal{F}}) + cost(\mathcal{T}_i^{\mathcal{B}})$
$\mathcal{P}_i^{\mathcal{F}} \leftarrow$ EXTRACTMINPATH(A_{i-1}, n, $\mathcal{F}$) ▷ If anchor exists
$\mathcal{P}_i^{\mathcal{B}} \leftarrow$ EXTRACTMINPATH(A_{i+1}, n, $\mathcal{B}$) ▷ If anchor exists
return $\mathcal{P}_i^{\mathcal{F}} \bigcup \mathcal{P}_i^{\mathcal{B}}$

procedure EXTRACTMINPATH(A_i, n, $\mathcal{D}$)
  $\mathcal{P}_i^{\mathcal{D}} \leftarrow$ PATHLOOKUP($\mathcal{T}_i^{\mathcal{D}}$, n) ▷ tree traversal from n to root
  if $\mathcal{D}$ == $\mathcal{F}$ **then**
    $\mathcal{P}^{\mathcal{D}} = \mathcal{P}^{\mathcal{D}} \bigcup$ EXTRACTMINIMUMPATH(A_{i-1}, n, $\mathcal{F}$)
  else ▷ ($\mathcal{D}$ == $\mathcal{B}$)
    $\mathcal{P}^{\mathcal{D}} = \mathcal{P}^{\mathcal{D}} \bigcup$ EXTRACTMINIMUMPATH(A_{i+1}, n, $\mathcal{B}$)
  end if
  return $\mathcal{P}^{\mathcal{D}}$
end procedure

Algorithm 2 CommitAnchorSet

Tree calculations post user adjustment of an anchor set:
Input: anchor set, A_i
COMMITFORWARD(A_i)
COMMITBACKWARD(A_i)

procedure COMMITFORWARD(A_i)
  $\{n\} \leftarrow getNodes(A_i)$
  $C_i^{\mathcal{F}} \leftarrow getCosts(\{n\}, \mathcal{T}_{i-1}^{\mathcal{F}})$
  $\mathcal{T}_i^{\mathcal{F}} \leftarrow$ COMPUTETREE($\{n\}$, $C_i^{\mathcal{F}}$)
  COMMITFORWARD(A_{i+1}) ▷ If anchor exists
end procedure
procedure COMMITBACKWARD(A_i)
  $\{n\} \leftarrow getNodes(A_i)$
  $C_i^{\mathcal{B}} \leftarrow getCosts(\{n\}, \mathcal{T}_{i+1}^{\mathcal{B}})$
  $\mathcal{T}_i^{\mathcal{B}} \leftarrow$ COMPUTETREE($\{n\}$, $C_i^{\mathcal{B}}$)
  COMMITBACKWARD(A_{i-1}) ▷ If anchor exists
end procedure
procedure COMPUTETREE($\{n\}$, C)
  InitTreeCalculation($\{n\}$, C) ▷ initialize costs to fl. anchor
  $\mathcal{T} \leftarrow$ RunOptimization() ▷ Dijkstra's, etc.
  return $\mathcal{T}$
end procedure