Human-computer interaction in medial image analysis

Olabarriaga, S.D.

Citation for published version (APA):
Chapter 3

A Framework for Interactive Segmentation Based on Cases of Failure of the Automatic Method*

"After salutation, observing me to look earnestly upon a frame, which took up the greatest part of both the length and breadth of the room, he said perhaps I might wonder to see him employed in a project for improving speculative knowledge by practical and mechanical operations. But the world would soon be sensible to its usefulness, and he flattered himself that a more noble exalted thought never sprang in any other man's head".


When putting visual data at the centre of information systems, one is bound to discover that segmenting the object of interest is a pivotal issue. In response, no other topic in computer vision has attracted so much attention as the segmentation of the image. Examples of advances obtained in segmentation methods are the inclusion of a priori knowledge (model-based segmentation), the integration of boundary- and region-based techniques and boundary optimisation based on the balance of image and shape constraints (e.g. deformable models). Despite this effort, in complex domains such as medical imaging, no automatic method is known to be generally applicable, completely reliable and robust. As a consequence, interaction is part of many segmentation procedures in practice, to bootstrap, steer or correct the automated segmentation method, and it is likely to remain so for many years to come.

In spite of that, interactive segmentation is a controversial topic. On the one hand, interaction has a poor reputation among people endorsing automated methods because it usually yields subjective results and it is a time-consuming task. On the

* Preliminary versions of this chapter were presented at the IPMI'97 and VISUAL'97 conferences.
other hand, to “put the user in the loop” is often suggested as a strategy to obtain reliability necessary for medical applications, e.g. in [27], [20], [9], [4] and [7]. There is an urge, thus, to integrate the advantages of computation and interaction to improve the results obtained with image segmentation methods.

In response to this demand, we put forward a Framework for Interactive Segmentation (FIS), an approach where interaction is motivated by the cases of failure of the automatic method (see also [19] and [26]). The information provided by the user in each case is used to steer parameters for the automatic part, enabling recovery from the failure condition. By integrating interaction and computation in such way, it is our aim to expand the application of segmentation into much more complex images; enhance the robustness, accuracy and reproducibility of results due to human-computer collaboration; and make the segmentation process more efficient due to maximised impact of interaction.

In this chapter we initially formulate the requirements posed on interactive segmentation methods (section 3.1). In section 3.2, we present the basic conception of the framework for interactive segmentation and discuss the main components: reasoning, interactive tools, visual feedback supporting interaction, and the backing computational method. In section 3.3 we present examples of how FIS is to be applied to two state-of-the-art paradigms for interactive segmentation methods, namely deformable models [17] and the active paintbrush [13]. A discussion about related work is presented in section 3.4.

3.1 Requirements for Interactive Segmentation

Little attention has been devoted to developing requirements that define the competence of interaction segmentation methods, with the consequence that the design of such methods is often approached in an ad-hoc manner. The requirements below are formulated on the basis of the survey about interaction in medical image segmentation presented in chapter 2. These requirements represent an effort to bring structure into the design of interactive segmentation methods, aiming at reliable, precise, predictable and reproducible object delineation with efficient interaction.

- **Requirement #1: Integrated Process.** Interaction and computation should be integrated into one process with continuous outcome and complementary roles for the user and the machine.

  Consider the following general formulation for a segmentation process:

  \[ R = S(g, \vec{p}) \]

  where \( S \) is a model-based computational method, \( g(x) : \mathbb{N}^D \rightarrow \mathbb{N} \) is the grey image, \( \vec{p} \in \mathcal{P} \) is a vector of parameters for the model contained in \( S \), from the set of all possible parameter values \( \mathcal{P} \), and \( R \subset \mathbb{N}^m \) is the result corresponding to the object delineation in the image grid. The segmentation model \( M \) describes about how the object is possibly depicted in the image based on prior knowledge about the application. An example: \( R \) is the contour in a bi-dimensional image,
$S$ is the deformable model [17] and $p$ corresponds to the initial curve and the weights of the shape and image constraints.

We assume that user intervention is necessary only when the result obtained with $S$ is off target:

$$R \neq \hat{R} \implies \text{Interaction},$$

where $\hat{R}$ is the target delineation.

Let $I \in \mathcal{I}$ denote the information provided by the user during interaction (e.g. image grid positions corresponding to corrections in $R$), from the set of all possible interactions $\mathcal{I}$. Integration of interaction and computation here means that $I$ is not included directly as a literal part of $R$, but used to steer $p$ as follows:

$$\tilde{p}_I = \Upsilon(I) \implies \tilde{p} = \tilde{p}_A \oplus \tilde{p}_I,$$

where $\tilde{p}_I$ are the parameter values configured on the basis of interaction $I$ by an interpreter function $\Upsilon : \mathcal{I} \rightarrow \mathcal{P}$, $\tilde{p}_A$ are the parameters without interaction, typically configured on the basis of prior knowledge about the segmentation problem, and $\oplus$ stands for an operation that combines parameter values, e.g. by replacing some values in $\tilde{p}_A$ by corresponding values in $\tilde{p}_I$.

This requirement is introduced to safeguard against discontinuities in the object delineation that could be introduced by simply gluing results obtained with two different processes, one automatic and the other interactive. Contour continuity aims at guaranteeing local contour properties, e.g. curvature, desired for the reliability of subsequent operations like measuring object shape or viewing its edge. Additionally, this requirement aims at less subjective results and efficient interaction, since the user participates in the process only when the computational method cannot find the desired result with the default parameter settings. As a consequence, the outcome of segmentation is objective and interaction is avoided everywhere else.

- **Requirement #2: No side-effects.** The impact of interaction should be restricted to the portion of the delineation that needs correction.

In a typical situation, only part of the delineation is wrong and requires correction, therefore interaction is localised:

$$R(T_I) \neq \hat{R}(T_I) \implies I[T_I],$$

where $T_I \subset \mathbb{N}^n$ represents a portion of the delineation $R$, and $I[T_I]$ indicates that the information $I$ provided with interaction is bound on $T_I$. An example: $T_I$ is an interval defining a curve segment and $I[T_I]$ corresponds to new contour positions in the interval $T_I$. 


The results obtained with and without interaction should be identical in the parts of the delineation that are not affected by interaction:

\[ I[T_I] \implies \forall B : B \cap T_I = \emptyset \implies R_I(B) = R_A(B), \]

where \( B \in \mathbb{N}^n \) stands for a portion of the delineation \( R \), \( R_I \) and \( R_A \) are the results obtained respectively with and without interaction.

To fulfill this demand, parameters for the computational method should have local control, with a confined influence on the segmentation result.

This requirement aims at localised and predictable impact of interaction in the segmentation result, in the sense that interaction should not affect significantly portions of the delineation that are already correctly placed.

**Requirement #3: Same intention, same result.** Segmentation results should be invariant under small differences in user intervention with the same intention.

Results should be reproducible under predictable limits of variation in the interaction when the target delineation is the same. Three distinct situations are identified:

A. Interaction is not needed to obtain the desired result \( \hat{R} \), however the user intervenes in the process anyway, typically with minor corrections. In this case, the results obtained with and without interaction should be identical:

\[
\left( \exists \vec{p}_A : S(g, \vec{p}_A) = \hat{R} \right) \land I \implies S(g, \vec{p}_A) = S(g, \vec{p}_A \oplus \vec{p}_I).
\]

B. Interaction is needed, but users\(^\dagger\) provide the information for correction in slightly different ways \( \{I_1, I_2\} \), leading to the computation of different parameters \( \{\vec{p}_1, \vec{p}_2\} \).

In this case, results obtained with interaction that is based on different information should be identical within arbitrarily small variations in \( \vec{p}_I \):

\[
|\vec{p}_1 - \vec{p}_2| \leq \epsilon \implies S(g, \vec{p}_A \oplus \vec{p}_1) = S(g, \vec{p}_A \oplus \vec{p}_2),
\]

where \( \epsilon \) typically corresponds to variations of \( I \) in respect to the image grid, such as the size of the window in which the image is displayed on the screen (scale) and the position and orientation of the input data (translation and rotation).

Note that case A reduces to case B with null \( \vec{p}_I \).

To fulfill this demand, the segmentation method must place \( R \) at an optimum of segmentation quality \( Q \) computed from the image data and the model:

\[
S(g, \vec{p}) = \max_{\{R \in \mathcal{R}\}} Q_{g, \vec{p}}(R),
\]

\( \dagger \)It could be one user in different segmentation sessions or different users.
such that the same result is obtained if the variation in $p$ is small enough to keep the result within the same optimum of $Q$. $R \subseteq N^n$ is the set of all possible results in the grid.

C. The users provide essentially the same information for correction in different order $\{I_1, I_2\}$ and $\{I_2, I_1\}$.

In this case, the final result obtained with different sequences of interaction should be identical:

$$\hat{p}_{I_1} \land \hat{p}_{I_2} \Rightarrow S(g, \hat{p}_A \ominus \hat{p}_{I_1} \oplus \hat{p}_{I_2}) = S(g, \hat{p}_A \ominus \hat{p}_{I_2} \oplus \hat{p}_{I_1}).$$

If $I_1$ and $I_2$ are spatially independent, i.e. $T_{I_1} \cap T_{I_2} = \emptyset$, this demand is fulfilled in a trivial way when the impact of interaction is local (see requirement #2). When $I_1$ and $I_2$ are spatially dependent, sequence invariance reduces to case B only if the combined effect of interaction can be expressed as a small variation in parameter values that leads to the same optimum of quality. Sequence invariance in other circumstances is more difficult to achieve.

Note that the ultimate outcome of the interactive process is subjective in nature, since the target delineation $\hat{R}$ depends on user judgement. This requirement, however, is introduced to ascertain reproducibility of the segmentation result even when human intervention is essential [27], reinforcing the integration of interaction and computation imposed by requirement #1.

- **Requirement #4: High Semantics.** A small action performed by the user should have big impact in the result towards the target object delineation.

To fulfill this demand, the tools used for interaction should perform high-level operations adequate to the different situations in which user intervention is necessary. Interaction is needed when the computational method cannot find the correct result based only on prior knowledge; in these situations, the user can provide information that is used to reconfigure the parameters for the computational method:

$$\forall \hat{p}_A : S(g, \hat{p}_A) \neq \hat{R} \Rightarrow \exists I : S(g, \hat{p}_A \ominus \hat{p}_I) = \hat{R}.$$

High semantic level of interaction means that this information is provided with specialised interactive tools with optimised behaviour to convey user intentions to the program in an intuitive way by means of minimal actions:

$$\forall I : S(g, \hat{p}_A \ominus \hat{p}_I) = \hat{R} \Rightarrow \exists T \in T_S : I = T(U_I),$$

where $T$ is a specialised interactive tool from a repertoire $T_S$ that depends on the computational method $S$, and $U_I$ represents user actions such as mouse clicks.

The underlying demand is that the user should be capable of immediately judging which tool is needed. Furthermore, the impact of interaction on the segmentation result should be predictable, where the resulting delineation makes
sense to the user after the parameters of the computational model have been reconfigured.

This requirement is introduced to guarantee efficient interaction by offering the user a variety of tools, each with a specific behaviour corresponding to different situations where intervention is needed. Tools with specific behaviour for different intention are expected to be more efficient to operate than an universal tool for all circumstances.

- **Requirement #5: Intelligence.** It should be possible to learn from user interventions.

The new configuration of parameters based on information provided by interaction should be reusable in future segmentation tasks in two different situations:

A. To segment another similar object in the same image:

\[
\hat{R}_1 = S(g_1, \overline{p}_A \oplus \overline{p}_l) \wedge d_R(\hat{R}_1, \hat{R}_2) \leq \epsilon \Rightarrow \\
\hat{R}_2 = S\left(g_1, \overline{p}_A \oplus \mathcal{L}(\overline{p}_l)\right),
\]

where \(d_R(.) \leq \epsilon\) indicates an arbitrarily small difference between the two segmentation results in terms of shape and image properties (e.g. the Hausdorff distance [2] and the sum of squared differences of the Laplacian [11]), and \(\mathcal{L}(\cdot)\) represents the learning procedure. Example: a modified background affects the appearance of all objects in the image in a uniform way, so the corrected knowledge for one object should hold for all the others.

B. To segment the same type of object in a similar image:

\[
\hat{R}_1 = S(g_1, \overline{p}_A \oplus \overline{p}_l) \wedge d_g(g_1, g_2) \leq \epsilon \Rightarrow \\
\hat{R}_2 = S\left(g_2, \overline{p}_A \oplus \mathcal{L}(\overline{p}_l)\right),
\]

where \(d_g(.) \leq \epsilon\) indicates an arbitrarily small difference between two images, e.g. the sum of squared differences of the Laplacian [11]. Examples: contiguous slices from a 3-D data set or images in the same study.

To fulfil this requirement, the representation of knowledge contained in the segmentation method should be structured and be flexible to permit dynamic adjustments. Examples of simple learning mechanisms are: (1) to use \(\overline{p}_l\) directly as \(\overline{p}_A\) in the next segmentation task and (2) to indirectly use \(\overline{p}_l\) to revise \(\overline{p}_A\) based on statistics about corrections motivated by interaction.

This requirement is introduced where it is desired to learn from user input, to improve the prior knowledge in the computational method and possibly reduce the need for user intervention, guaranteeing efficiency in continued interaction.
3.1.1 Summary

The requirements for interactive segmentation are summarised in tab.3.1. Note that requirements #2 and #3 are demands to produce a trustworthy result, #4 and #5 are motivated by efficiency, and requirement #1 aims at both. In addition to these, three practical demands are posed on the interactive method: (1) it should be flexible to deal with the diversity of segmentation problems found in the reality of medical applications; (2) the final segmentation result should be always confirmed by the user to guarantee credibility necessary for clinical applications [7]; and (3) the method should have performance compatible with interactive response.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Integrated process</td>
<td>Reliability + Precision</td>
</tr>
<tr>
<td></td>
<td>Efficiency</td>
</tr>
<tr>
<td>2 No side-effects</td>
<td>Predictability</td>
</tr>
<tr>
<td>3 Same intention, same result</td>
<td>Reproducibility</td>
</tr>
<tr>
<td>4 High semantics</td>
<td>Efficiency (short term)</td>
</tr>
<tr>
<td>5 Intelligence</td>
<td>Efficiency (long term)</td>
</tr>
</tbody>
</table>

Table 3.1: Summary of the requirements for interactive segmentation and their purpose in the context of medical image analysis.

A study of the existing interactive methods for medical image segmentation presented in chapter 2 revealed that most of them do not fulfil the requirements above. A summary of the most promising methods is presented in tab.3.2, supporting the conclusion that interactive segmentation methods that completely fulfil these requirements are not found in the existing literature.

<table>
<thead>
<tr>
<th>Method</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probabilistic hierarchy [5]</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>Live wire [6],[3]</td>
<td>+ + + +</td>
</tr>
<tr>
<td>Constrained hierarchy [8]</td>
<td>+ + +</td>
</tr>
<tr>
<td>BASOC [16]</td>
<td>+ + +</td>
</tr>
<tr>
<td>Ziplock snakes [18]</td>
<td>+ + +</td>
</tr>
</tbody>
</table>

Table 3.2: Fulfilment of selected methods to the requirements for interactive segmentation.
3.2 Architecture of a Method Based on the Framework for Interactive Segmentation

In the framework for interactive segmentation (FIS), the task is the **delineation** of objects that are expected to be represented on the image in a given way. We assume that the user knows the location of the correct delineation, the only problem being to convey this information to the computer in an efficient way.

In response to the requirements formulated in section 3.1, FIS proposes the unification of the computational and interactive parts into a single integrated segmentation process based on the conception illustrated in figure 3.1.

Interaction takes place in a scenario where the result is progressively refined by alternating actions performed by the computational method and the user. The backbone of the process is a computational method based on a segmentation model with prior knowledge about how the object is ideally depicted in the image. For efficiency, this method is initialised on the basis of information provided by the user or based on prior knowledge about the segmentation problem. After initialisation, the results obtained with this method are displayed on the screen for user evaluation. According to requirement #1, user interventions are necessary only when the result generated by the computational method is wrong, that is, when the method “fails.” In such cases, the user employs interactive tools to provide information indicating the adequate correction. The data input by the user feeds a reasoning procedure that reconfigures the parameters for the segmentation model to obtain the desired segmentation result.

In the next sections we present considerations about the design of each component of FIS: the reasoning process, based on the cases of failure that motivate interaction...
(section 3.2.1), the interactive tools providing the information needed to reconfigure the method in each case (section 3.2.2), the visual feedback supporting the evaluation of the situation at hand and user intervention (section 3.2.3) and the computational method backing the whole process (section 3.2.4). Since we do not want to commit FIS to any particular computational method, segmentation model or user interface style, we keep the discussion to an abstract level, raising relevant aspects on the basis of reasonable assumptions about a general setting.

### 3.2.1 Reasoning about Failure

In this section we discuss the interpretation of interaction in a scenario where user interventions are motivated by failure of the computational method. We characterise the situations in which a model-based segmentation method can fail, identifying a limited number of cases for user intervention (see figure 3.2).

![Figure 3.2: The learning component of FIS, based on a limited number of cases where the computational method may fail.](image)

Each case demands different strategies to reconfigure the computational method based on information provided by the user. In the context of this work, the function responsible for interpreting user actions into parameter values for the computational method is defined as follows:

$$\tilde{P}_I = Y(c, I_c),$$

where $c \in [1, N_c]$ represents a case for failure from a limited set of cases that depend on the computational method $S$, and $I_c$ is the information provided by the user in each case.

In the sections that follow we discuss the reasons why a computational method may fail, characterising general cases and the type of information necessary to recover from a failure situation.
Why a Computational Methods Fails

The computational method fails when the object in the image deviates too much from the ideal appearance covered by the segmentation model. In medical images, objects may appear differently from expected due to pathology, imaging artefacts, noise, abnormal anatomy, low radiation dose, short acquisition time to prevent patient discomfort, and alike. Irrespectively to the real cause, “abnormal” appearance corresponds to an altered pattern in the image data, observed in terms of the image intensity, the object shape, or both. Problematic situations occur when patterns deviate too much from the “normality” covered by the segmentation model and cannot be handled by the computational method, therefore variations due to biological diversity or image acquisition conditions may cause failure as well. It is important to note that, in most cases, failure is probably confined to a portion of the object. As a consequence, the model is valid for most part of the object, but it needs to be locally corrected, tuned or steered on the basis of interaction to capture the object in the image.

To determine the situations in which a method fails, one must consider the limits of deviation from normality that can be handled by the model components; outlying features are likely to cause failure. The outcome of this case-by-case analysis is a finite number of cases of failure motivating interaction in the context of a particular model and method. As a consequence, there is no universal list of cases of failure of all possible methods. However, most segmentation methods are bounded to fail in three general situations: (1) when the visual evidence of the object delineation is too low or absent, (2) when the visual evidence is present, but the method is disturbed by another object in the image and (3) when the object in the image deviates from the knowledge in the segmentation model. The general cases discussed in the sequel serve as a guide to determine the situations in which a particular method fails.
Case #1: Low or absent visual evidence.

In this case, neither the user nor a computational method locally observes supporting evidence of the object in the image. This may happen when an alien object blocks the view (e.g. a piece of metal), noise is too heavy (e.g. nuclear medicine) or neighbouring tissues respond to imaging in a similar way (e.g. touching organs in the belly) – see figures 3.3 and 3.4.

![Illustration for the case of low or absent visual evidence.](image)

In these situations, the image data alone do not provide enough information to clearly distinguish the structure from the background, so contextual knowledge must be added locally to fill in the missing information and complete the object delineation in an indirect way. The adequate correction in this case is to locally reduce the importance of the image-derived component, while emphasising *a priori* global shape assumptions and constraints.
Case #2: Wrong visual evidence.

In this case, the human observer may see the proper visual evidence, since it is present in the image, but the computational method confuses it with interfering information from the neighbourhood. This may occur when there is a neighbouring structure with a stronger edge, such as a tumour, calcifications or other organs in the belly - see figures 3.5 and 3.6.

Figure 3.5: Illustration for the case of wrong visual evidence. (a) A neighbouring object has similar intensity pattern, but the edge is still visible. (b) Internal object.

Figure 3.6: Examples of wrong visual evidence - courtesy of the Image Sciences Institute, University Medical Center Utrecht. Slice of an abdominal CT showing where the delineation of the aortic aneurism can be disturbed by stronger edges corresponding to calcifications and to neighbouring structures like the spine.

In this case, there is visual evidence of the object and the computational method can detect it; however, the method chooses the evidence corresponding to the wrong object. Extra information is therefore needed to help the method pick the correct alternative and successfully find the target delineation. To recover from failure, the computational method should confine the admissible results to the zone indicated by the user, disregarding visual evidence that is far from the target, and increase the importance of the global shape constraint.
Case #3: The object in the image does not fit the model.

In this case, visual evidence is clear, but the object does not fit the segmentation model in the computational method. In these situations, the model has to be replaced locally to achieve the desired object delineation.

Case #3.A: Unseen visual evidence.

In this case, the user perceives an edge, but the computational method cannot detect it because the intensity pattern in the image differs from what has been incorporated in the computational model. This situation may happen if the object is placed against an unusual background, or if the imaging conditions change the object intensity pattern drastically - see examples in figures 3.7 and 3.8.

![Figure 3.7](image1.png)

Figure 3.7: Illustration for the case of unseen visual evidence. (a) A touching object provides an inward- rather than an outward-pointing gradient at the boundary. (b) A non-uniform intensity pattern causes variations in the object/background contrast along the boundary.

![Figure 3.8](image2.png)

Figure 3.8: Example where the joint space in osteoarthritic ankles appears differently in two X-ray images of the same study - courtesy of the Image Sciences Institute, University Medical Center Utrecht.

In this case, the type of image property used in the computational model must be replaced locally. For the example in figure 3.8, rather than using a detector of step-edges, the method should locally use a detector of bright lines in the image to follow the proper contour line.
Case #3.B: *Distortion from the local geometry.*

In this case, the object in the image violates constraints imposed by the local shape model and cannot be captured by the computational method. This situation may happen when the object in the image has less detail than expected from the model (e.g. less resolution due to partial volume effect) or, reversibly, the image shows more detail than expected (e.g. due to pathology) – see figures 3.9 and 3.10.

![Figure 3.9: Illustration for the case of distortion from the local geometry. (a) The object has less detail than expected in the right part. (b) The object has more detail than expected.](image)

![Figure 3.10: Two slices of the same abdominal CT showing different level of detail for the aortic lumen - courtesy of the Image Sciences Institute, University Medical Center Utrecht. (a) The usual appearance: smooth boundary. (b) The boundary has bumps due to an aneurism.](image)

In this situation, the local shape model contained in the computational method must be modified locally to permit the delineation to follow the existing visual evidence with more freedom. For example, the smoothness constraint in the model could be relaxed locally to permit a jagged boundary, or degrees of freedom could be added to the curve representation locally.
3.2. The Architecture of a Method Based on FIS

Case #3.C: Distortion from the expected global geometry.

In this case, the object in the image deviates from the global shape model and cannot be captured by the segmentation method. This situation may happen when a geometric model is available for the normal case, but the abnormal case is hard to catch geometrically, such as when a part of the object is missing (e.g. a part of the organ was removed by surgery) or severely distorted (e.g. the organ is growing a polyp), or the object is divided into pieces (e.g. a broken bone) – see figures 3.11 and 3.12.

Figure 3.11: Illustration for the case of distortion from the global geometry. (a) A part is missing. (b) The object has an appendix. (c) The object is broken.

Figure 3.12: Two slices of the same CT of the head showing a case for distortion from the usual global shape. (a) The normal situation. (b) The skull is interrupted as a consequence of a surgical intervention. Data provided by the Image Sciences Institute, University Hospital Utrecht, The Netherlands.

In this case, the global shape component of the segmentation model should be locally replaced to enable the computational method to find the correct delineation. The simplest type of correction is to locally ignore the global shape component, enabling the method to freely follow the existing visual evidence of the object. Another alternative would be to locally replace the global shape model in the method by the curve drawn by the user. Note that in the case of objects divided into pieces, the correction strategy must account for changes in topology.
Summary

Table 3.3 presents a summary of the cases of failure of model-based segmentation methods and examples of modifications to the model that could be applied to obtain the proper segmentation result.

<table>
<thead>
<tr>
<th>Case</th>
<th>Cause</th>
<th>Model correction</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Low or absent visual evidence</td>
<td>Ignore visual evidence</td>
</tr>
<tr>
<td>2</td>
<td>Wrong visual evidence</td>
<td>Confine result to narrow region</td>
</tr>
<tr>
<td>3.A</td>
<td>Unseen visual evidence</td>
<td>Replace image feature model</td>
</tr>
<tr>
<td>3.B</td>
<td>Local Shape Distortion</td>
<td>Modify local shape description</td>
</tr>
<tr>
<td>3.C</td>
<td>Global Shape Distortion</td>
<td>Replace global model</td>
</tr>
</tbody>
</table>

Table 3.3: Summary of the different types of cases where computational methods can fail and the corresponding model corrections.

These cases are general and can serve as guidelines to determine in what essentially different ways a specific segmentation method fails. These guidelines are used in chapter 6 to develop an interactive segmentation solution for a medical application.

The identification of cases of failure is useful to indicate the circumstances in which interaction is needed, the applicable model correction, and the type of information to be provided by the user in each case. Structuring interaction into cases of failure provides a basis for the design of tools that can learn from user interaction. The level of reasoning implemented in such tools depends on the amount of contextual knowledge available. In the simplest setting, the user indicates the type of failure, and the corresponding correction is applied to locally steer the model of the object at hand, in a one-time fashion. In a more sophisticated implementation of this framework, a higher-level mechanism could be used to update the prior knowledge in the method based on the corrections resulting from interaction, leading to long-term learning (e.g. with machine learning techniques [21]). And finally, in a truly intelligent system, the information provided by the user would be used not only for model correction, but also to determine the type of failure at hand, e.g. using case-based reasoning [1].

In conclusion, the implementation of reasoning about interaction based on the types of cases discussed here address the requirements #1 (integrated process), #2 (no side-effects), #4 (high impact) and #5 (intelligence).

### 3.2.2 Interactive Tools

The interactive tools available to the user must be powerful, intuitive and allow efficient interaction. To avoid repetition of general principles for the design of user interfaces presented in [22], here we focus on aspects that are relevant to support FIS.

Apart from the indication to bootstrap the process in an efficient way, interaction is essentially motivated by failure of the computational method, such as to steer
computation. As such, the interactive tools should enable the user to efficiently notify the computer about the part of the delineation that is incorrect, the possible type of failure \((c)\) and the necessary reconfiguration of model parameters \((I_c)\). Since the cases discussed in section 3.2.1 require specific information, each case \(c\) should be addressed by an interactive tool \(T_c\) with specialised behaviour (see figure 3.13), defined as follows:

\[
T_c(U_I) \rightarrow \{c, I_c\}. 
\]

We assume that the user is capable of instantly deciding upon the tool to use from the visual feedback displayed on the screen. To facilitate the choice of tool for the failure at hand, the repertoire of interactive tools should be presented to the user in the form of iconic menus, where the icons depict the visual configurations characterising each type of failure. In a direct manipulation setting, the user picks the “wrong visual evidence” tool from the menu (indicate the type of failure), drops it on a contour part that needs correction, and edits the contour locally (indicate where and how the contour should be repaired). Similarly, the user picks the tool “unseen visual evidence,” drops it on the wrong part, edits the contour locally, and chooses another type of image feature from an iconic menu. Note, however, that one-to-one correspondence between tools and cases of failure is neither necessary nor desired in a truly intelligent setting, where the identification of the case at hand would be done automatically by the system.

In conclusion, the specialised interactive tools proposed here support local interaction at a high semantic level, providing information for higher levels of reasoning. As such, they address the following requirements for FIS: \#1 (integrated process), \#4 (high semantics) and \#5 (intelligence).

### 3.2.3 Visual Feedback

The image on the screen provides the user with all the necessary information for instant evaluation of the situation at hand. This information should enable the user to
evaluate whether the current delineation is correct and, when it is wrong, to determine the case for failure and to choose the adequate interactive tool.

In all circumstances, the position of the current delineation $R$ should be displayed on the image grid. With this type of information, the user can determine whether the delineation is correct and, if not, which portion needs to be corrected ($T_f$).

Visualisation of the current delineation is usually sufficient to detect the cases of failure referring to problems in the image data: case #1 (absent or low visual evidence) and case #2 (wrong visual evidence). More information must be available to the user to select among cases that depend on the segmentation model adopted by the computational method (#3.A, #3.B and #3.C). For these cases, the user must also receive visual feedback about the model adopted by the computational method - see figure 3.14.

![Diagram](attachment:image.png)

**Figure 3.14:** Examples of the visual feedback presented on the screen for a boundary-based method, showing the current delineation $R$ and various visualisations of the segmentation quality (see text).

To reveal the segmentation model to the user, we adopt a measure of segmentation quality $Q_{g,p}(R)$. This measure indicates the similarity between the delineation $R$ and the ideal model defined with parameter settings $\tilde{p}$. Large values of $Q$ show that the delineation is similar to the ideal, while small values indicate deviation. For simplicity, in the discussion below we assume that $Q$ can be computed for all pixels, for any portion of $R$ and that $Q$ can be computed separately for each of the model components.

Four types of information should be displayed to the user:

- The position of the current delineation in the image.

- The local quality of the current delineation with respect to the current model parameter settings:

$$Q_{g,\tilde{p}_m}(R(t)),$$

where $t$ is a parameter to navigate along the contour or region and $\tilde{p}_m$ refers to parameters for the complete model of a model component.
The quality of segmentation is displayed locally for each pixel along the contour or inside region $R$.

This type of visualisation is useful (1) to call attention to portions where $Q$ is low and possibly need user intervention and (2) to support discrimination between the cases #3.A, #3.B and #3.C, by indicating which of the model components reports low quality.

- **The local segmentation quality in the image grid:**

  $$Q_{g,\tilde{p}_m}(x),\ x \in \mathbb{N}^D,$$

  where $x$ represents the pixel coordinates.

  The quality of segmentation is displayed for each pixel in the image $g$.

  This type of information is useful (1) to reveal what type of object the computational method is looking for, facilitating the identification of cases #3.A, #3.B and #3.C and (2) to call attention to alternative image positions where the delineation could be located.

- **The gradient of the local segmentation quality of the current delineation in respect to the image grid:**

  $$\nabla_x Q_{g,\tilde{p}_m}(R(t))$$

  The gradient of the segmentation quality is displayed as vectors along the contour or inside region $R$, indicating the direction of local deformation of the current delineation to obtain smaller deviation from the model. If these vectors do not point towards the desired result, the model parameters are wrong and user intervention is needed.

  This type of information is useful (1) to help the user predict the behaviour of the computational method and (2) to reveal the influence of the model components separately, facilitating the identification of the type of case at hand (#3.A, #3.B and #3.C).

Visual feedback about the segmentation process should be presented with intuitive visual metaphors that are overlaid on the grey image, following recommendations in [12] and [22]. Some examples are presented below (see also figure 3.14): (1) represent the object delineation by a curve or a region, using a different colour for parts that deviate too much from the model or highlighting parts that have changed position since the last user intervention or computation. (2) highlight regions in the image that match the intensity profile defined by the image feature adopted in the model; and (3) show how the computational method would deform the current contour by arrows. The screen should be updated in real-time during interaction to help the user predict the impact of interactive tools on the segmentation results.

In summary, the type of information shown on the screen determines the type of reaction that can be expected from the user via the interactive tools. Following this line of thought, it might seem appealing to reveal more aspects of the automatic
method and offer more interactive tools as a strategy to increase the help that could be provided by the user. One must be aware, however, that interaction must be kept at a simple and intuitive level if the system is targeted for medical users. An adequate balance in this case is fundamental to guarantee that the information needed for successful segmentation is displayed, without overloading the user with irrelevant details.

In conclusion, the type of visual feedback suggested here enables the user to determine when intervention is needed, the portion of the delineation that needs correction, and the type of case at hand, therefore supporting requirements #1 (integrated process) and #4 (high semantics).

### 3.2.4 The Computational Method

We assume that the computational method $S$ is based on a model with prior knowledge about how the object is ideally depicted in the image. This model integrates different sources of data such as image intensity properties, local and global geometry, expected position and orientation in the image grid. The role of the segmentation model is to provide contextual knowledge to increase the chances of success of the computational method, particularly when the delineation seen by the human operator does not have supporting evidence in the image data.

To support the necessary visual feedback for the user, as discussed in section 3.2.3, the computational method should generate not only a segmentation result ($R$), but also indicate the quality of this result $Q$ in terms of the current model settings (see figure 3.15).

Besides providing support for visual feedback, the computational method must comply with the following conditions to be used as the backbone of FIS:

- Be steerable by means of parameters $p$ that can be dynamically reconfigured on
the basis of interaction:
\[ \vec{p} = \vec{p}_A \oplus \vec{p}_I. \]

- Be locally controllable, that is, the influence of parameters \( \vec{p} \) is limited to a part of the delineation \( T \):

\[
R = S(g, \vec{p}) \land R_1 = S(g, \vec{p}_1 \oplus \vec{p}_I \left| T_1 \right) \implies R_1(T) = R(T), \forall T : T \cap T_1 = \emptyset,
\]

where \( \vec{p}_1 \left| T_1 \right. \) indicates that the parameters \( \vec{p}_1 \) are bounded on \( T_1 \).

- Be invariant to small modifications in the parameter values motivated by interaction, displaying non-chaotic behaviour:

\[ |\vec{p}_1 - \vec{p}_2| \leq \epsilon \implies S(g, \vec{p}_1) = S(g, \vec{p}_2). \]

- Be flexible to accommodate different sources of knowledge about the segmentation problem that are originated from the image data, expected geometry and user interaction. This is achieved by describing the appearance of the ideal object with a combination of image and shape features \( F_i \) chosen from providing a varied repertoire of features \( F_i \in \{ F_1, F_2, \ldots, F_{N_f} \} \).

- Produce predictable output within an ensemble of solutions \( R \), accounting for acceptable differences between the segmentation result and the ideal model, such that:

\[ d_R(R, R_M) \leq \Delta_R, \]

where \( R_M \) represents the ideal result defined by the segmentation model \( M \), and \( \Delta_R \) stands for the acceptable difference determined based on the variation observed in the available delineation samples or other type of prior knowledge.

In conclusion, a computational method conforming with the conditions discussed above provides support for the implementation of visual feedback, interactive tools and reasoning as described respectively in section 3.2.3, section 3.2.2 and section 3.2.1, fulfilling all the five requirements for FIS. For an example of such a method, see the deformable model described in chapter 4.

3.2.5 Summary

Figure 3.16 contains a summary of the framework for interactive segmentation. Note that the initialisation and prior knowledge components have been left out of the discussion because they depend too much on the technical aspects of the segmentation technique underlying the computational method.
3.3 Application of FIS to State-of-the-Art Segmentation Methods

As an exercise, in this section we discuss how the framework for interactive segmentation is to be implemented on the basis of two state-of-the-art paradigms that already incorporate interaction: deformable models [17] and active paintbrush [13].

3.3.1 FIS Based on the Deformable Model Paradigm

Deformable models represent a class of segmentation methods in which an initial curve $R_0$ is deformed on the basis of integral constraints upon the object’s boundary (see details in [17]). These constraints define the boundary model $M$ for an ideal situation. The goal of deformation is to bring the contour $R$ in the image as close as possible to $M$, respecting the actual configuration of the data while balancing shape and image constraints.

Consider the implementation of the FIS framework where the computational method is a deformable model defining a smooth boundary - small curvature $\kappa(t)$ - located at discontinuities in the image intensity function - large gradient magnitude $\|\nabla g(x(t), y(t))\|$. For simplicity, the computational method $S$ is formulated as
follows:

$$
\min_{\{R \in \mathcal{R}\}} \Theta(R),
$$

with

$$
\Theta(R) = \int_{t} W_K(t) \kappa^2(t) + W_g(t) \left\| \nabla_{\sigma(t)} g(x(t), y(t)) \right\|^{-1} dt,
$$

where \( t \) is the path parameter for navigation along the curve \( R \) representing the contour, \( W_K(.) > 0 \) and \( W_g(.) > 0 \) are the weights of the shape and image constraints, and \( \sigma(.) \) is the aperture used to compute image derivatives. Note that \( W_K(t), W_g(t), \) and \( \sigma(t) \) are dependent on the path parameter to allow for local steering.

The process is initialised with a curve \( R_0 \) provided by the user and with \( W_K, W_g \) and \( a \) determined on the basis of prior knowledge about the segmentation problem. The initial curve is deformed and the result is displayed on the screen for user evaluation with metaphors illustrated in figure 3.14. The quality of segmentation is given by:

$$
Q(R) = \frac{1}{\Theta(R)}.
$$

Situations where the result is not correct characterise failure in the context of FIS. In such situations, the deformable model is corrected on the basis of information provided by the user with the interactive tools, the curve is deformed again, and the new result is displayed for user evaluation.

A resulting contour \( R \) can be locally wrong due to the reasons discussed in section 3.2.1. The cases of failure and the corresponding correction of the deformable model are presented below:

- **Case #1:** Low or absent visual evidence in a curve segment.

  Two corrections are necessary:

  (1) Due to uncertainty in the data, reduce the weight of the image constraint \( W_g(t) \) in the curve segment:

  \[
  W_g^*(t) = \begin{cases} 
  w : & W_g(t) : t \in [t_1, t_2], \ \text{with} \ 0 < w < 1 \\
  W_g(t) : & \text{otherwise}, 
  \end{cases}
  \]

  where \( W_g^* \) is the locally modified weight for the image feature, \( w \) is a factor determined from prior knowledge, and \([t_1, t_2]\) is the path interval corresponding to the curve segment affected by interaction.

  (2) When the data provides little evidence of the boundary location, add a local constraint to the model that attracts the contour \( R \) to a curve \( R_I \) indicated by the user, employing a modified objective function \( \Theta^* \):

  \[
  \Theta^*(R) = \Theta(R) + \Delta(R, R_I)
  \]

  where \( \Delta(.) \) measures the Euclidean distance between \( R \) and \( R_I \) as follows:

  \[
  \Delta(R, R_I) = \int_{t \in [t_1, t_2]} W_{\Delta}(t) \left\| R(t) - R_I(t) \right\| dt.
  \]
Chapter 3. A Framework for Interactive Segmentation

$R(t)$ and $R_i(t)$ are corresponding points in the curves and $W_\Delta(t) > 0$ is the weight of the extra constraint.

- Case #2: Wrong visual evidence in a curve segment.

Two corrections are necessary in this case:
1. To localise the correct evidence for the boundary, reduce the scale used to compute the image gradient in the curve segment

$$o^*(t) = \begin{cases} w \cdot \sigma(t) & t \in [t_1, t_2], \text{ with } 0 < w < 1 \\ \sigma(t) & \text{ otherwise} \end{cases},$$

where $\sigma^*$ is the locally modified aperture.
2. When the evidence is still misleading, add a local constraint to attract the boundary to the curve $R_i$ indicated by the user and proceed as in case #1.

- Case #3.A: Unseen visual evidence, i.e. the visual evidence of the boundary goes unnoticed by the computational method.

The adequate correction consists of replacing locally the image feature by another type $\Phi$ indicated by the user. The modified objective function becomes:

$$\Theta^*(R) = \Theta_\Phi(R) + \Theta_\nabla(R)$$

where the terms $\Theta_\Phi(R)$ and $\Theta_\nabla(R)$ have localised influence on the contour, respectively in the part affected and not affected by interaction:

$$\Theta_\Phi(R) = \int_t W_\kappa(t) k^2(t) + W_g(t) \| \Phi_{\sigma(t)}(x(t), y(t)) \|^{-1} dt, t \in [t_1, t_2]$$

$$\Theta_\nabla(R) = \int_t W_\kappa(t) k^2(t) + W_g(t) \| \nabla_{\sigma(t)}g(x(t), y(t)) \|^{-1} dt, t \not\in [t_1, t_2].$$

- Case #3.B: Distortion of the local geometry, i.e., the visual evidence of the boundary makes clear that the smoothness constraint in the model overreacts.

In this case, the importance of the shape constraint $W_\kappa(t)$ is reduced locally:

$$W^*_\kappa(t) = \begin{cases} w \cdot W_\kappa(t) & t \in [t_1, t_2], \text{ with } 0 < w < 1 \\ W_\kappa(t) & \text{ otherwise} \end{cases},$$

where $W^*_\kappa$ is the locally modified weight for the shape model.

Note that distortion of global geometry (case #3.C) does not apply here, since no global model is adopted by the computational method.
Four interactive tools are necessary to address each one of the cases above, with the following behaviour:

- **Tool #1**: to implement the corrections for case #1, the user must specify the curve interval \([t_1, t_2]\) where the correction applies and a curve \(R_f\) used to attract the contour. This information is provided by directly editing the current contour with the mouse, placing it precisely at the desired boundary position.

- **Tool #2**: corrections for case #2 also require the interval \([t_1, t_2]\) where it applies and a curve \(R_f\) to attract the contour. This information is provided by directly manipulating the current contour with the mouse, placing it roughly at the boundary position.

- **Tool #3.A**: for case #3.A, the curve interval \([t_1, t_2]\) where the correction applies and another type of image feature \(\Phi\) are needed. To provide this information, the user chooses \(\Phi\) from a menu and points out the curve interval with two mouse clicks.

- **Tool #3.B**: to correct for case #3.B, the user just has to indicate the curve interval \([t_1, t_2]\) where the curve is more jagged than normal with two mouse clicks.

In all cases, the reduction factor \(w\) used to correct the deformable model parameters is determined automatically on the basis of contextual knowledge. The curve modified by the user is deformed based on the new parameter settings, guaranteeing the integration of computation and interaction.

Practical issues on the side of the computational method hamper the implementation of this straightforward combination of deformable models and FIS. According to the description above, this implementation assumes that the boundary model admits local constraints, that these can be dynamically added and replaced to the model, and that their balance can be determined in a systematic manner with predictable results. Unfortunately, these assumptions are not true for the large majority of existing deformable model methods, with the consequence that they cannot be used in natura for the implementation of interactive segmentation as it would be expected (e.g. in [26]). This difficulty motivated the development of a new deformable model described in chapter 4, which was used in the implementation of FIS as suggested here - see chapter 5.

### 3.3.2 FIS Based on the Active Paintbrush Paradigm

The active paintbrush is an interactive segmentation technique based on the split-and-merge conception [10], where the “split” is done by the computational method and the “merge” is done interactively. In the original method described in [14], the image is initially over-segmented with the watershed transform [29]. The basins constitute primitive regions that are merged into a hierarchy supporting an efficient interaction process. In a subsequent step, the user selects with the mouse the region primitives that belong to the object, merging them to generate the final segmentation result.
Consider the implementation of FIS supported by a computational method that partitions the image \( g \) into a set of \( k \) primitive regions \( r_1, r_2, \ldots, r_k \), such that:

(i) \( g = \bigcup_{i=1}^{k} r_i \) (complete),
(ii) \( r_i \cap r_j = \emptyset, i, j \in [1, k] \) (non-overlapping),
(iii) \( \forall i \in [1, k] : r_i \) is connected.

The primitive partitioning of the image is denoted \( K \) and obtained as follows:

\[
K = S(F, n)
\]

where \( S \) is the watershed transformation, \( F = ||\nabla g|| \) is a feature image used as the basis for partitioning, with ridges corresponding to the edges of primitive regions \( r_i \), and \( n \) is the minimum number of pixels in a primitive region. To allow for local control needed to implement corrections in a method based on FIS, the computational method \( S \) has been modified as follows:

\[
K = S^*(K_0, F, n, T)
\]

where \( K_0 \) is an initial partitioning to be refined by \( S \) and \( T \subset \mathbb{R}^2 \) specifies the region for local refinement, outside which the partitioning given by \( K_0 \) is not affected.

The process is initialised by generating a partitioning of the image with the following settings: \( T \) includes the whole image, \( \sigma \) and \( n \) are determined from prior knowledge about the segmentation problem. This initial partitioning is displayed to the user, who can select primitive regions with the mouse and include them into the result. The segmentation result \( R \) is progressively built as an union of existing primitives \( r_i \):

\[
R = \bigcup_{i \in K_I} r_i, r_i \in K,
\]

where \( K_I \) is the set of regions interactively selected by the user.

The partitioning of the image is wrong when the user wants to select only a subset of a primitive region. This situation occurs when an edge is missing, characterising "failure" in the context of FIS. In such situations, user intervention is necessary to locally correct the partitioning, the new partitioning is displayed on the screen, and the selection of primitive regions by the user proceeds.

A partitioning \( K \) may be wrong due to the causes of failure discussed in section 3.2.1. The correction for each case consists of refining \( K \) in the region indicated by the user with modified parameters derived from interaction in the following way:

- Case #1: Low or absent visual evidence in a part of the boundary.

To compensate for a locally flat landscape in the feature image, an artificial ridge is added to \( F \) at the boundary position indicated by the user. The modified feature image \( F^* \) is obtained as follows:

\[
F^*(x, y) = \max \left( F(x, y), F_I(x, y) \right),
\]
where $F_l$ is the image containing the artificial ridge:

$$F_l(x, y) = U_l(x, y) * G_\sigma,$$

where $U_l(x, y)$ is an indicator function for the pixels pointed by the user and $G_\sigma$ is a Gaussian smoothing filter with the same scale $\sigma$ used for $F$.

- **Case #2:** Wrong visual evidence in a part of the contour.

  Two corrections are necessary in this case:
  
  1. To locate the visual evidence corresponding to boundary of interest, the aperture used to compute the feature image is locally reduced. The modified $F^*$ is generated as follows:

     $$F^*(x, y) = \left\{ \begin{array}{ll} \|\nabla u_{\sigma}(x, y)\| & : (x, y) \in T_l, \text{ with } 0 < w < 1 \\ F(x, y) & : \text{otherwise} \end{array} \right.,$$

     where $w$ is the reduction factor determined from prior knowledge and $T_l$ is the region affected by interaction.

  2. To account for small regions separating objects, locally relax the constraint on the minimum region size.

- **Case #3.A:** Unseen visual evidence, that is, the visual evidence is not captured by the type of image feature used by the computational method.

  In this case, the feature image is locally replaced by another type $\Phi$ indicated by the user, generating a modified $F^*$ as follows:

  $$F^*(x, y) = \left\{ \begin{array}{ll} \Phi_{\sigma}(x, y) & : (x, y) \in T_l \\ F(x, y) & : \text{otherwise} \end{array} \right..$$

Since no shape feature are used in the model built into the computational method, the cases #3.B and #3.C (distortion from local and global geometry) do not occur here.

Three tools are needed to interactively split the region in each of the cases of failure:

- **Tool #1:** to correct for case #1, the boundary position $U_l(x, y)$ is required. The user precisely indicates the position of the boundary by dragging the mouse directly over the image, and the trace of the mouse $U_l(x, y)$ is used to build the artificial ridge on the feature image $F_l(x, y)$.

- **Tool #2:** case #2 requires the specification of the region $T_l$ where the correction applies. The user indicates the region by dragging the mouse roughly over the position of the boundary. The convex hull of the mouse trace is used to derive $T_l$.

- **Tool #3.A:** case #3.A also requires a region $T_l$, as well as another type of image feature $\Phi$. To provide this information, the user roughly drags the mouse over the boundary position and picks $\Phi$ from an iconic menu. The region $T_l$ is derived from the mouse trace.
In all cases, the other parameters needed for correction are determined based on prior knowledge about the application. After the correction is applied, the modified watershed is executed again, guaranteeing integration of interaction and computation.

The combination of the active paintbrush paradigm with FIS proposed here tackles the problem of wrong image partitioning in paintbrush-like methods in a simple but effective way. As in the example based on deformable models, practical issues remain as obstacles for the implementation of this method, namely the addition of local control to the watershed transform and the modification of the feature image without introducing discontinuities.

### Related Work

Our work is based on the idea that an automatic method fails when the underlying segmentation model is inadequate to capture the object in the image, and that, in these situations, the user can provide contextual information that helps the method to prevent or recover from failure. This idea is partially found in the literature in different ways, as discussed below.

**The automatic method can fail because the model is wrong.**

In the early work described in [24], segmentation is composed of three main tasks: (1) decide upon what to look for - the Appearance Model Expert, e.g. the user; (2) choose the proper segmentation technique - the Operator Expert; and (3) choose the correct parameters for the technique - the Adaptive Operators. These components interact, automatically adjusting parameters based on differences between the results of computation and the goal at each level. *Errors* occur due to uncertainty regarding the appearance of objects in the image, for example, when an object that usually appears as a single region in the image is merged with other regions, fragmented into several regions, or both [23].

An analogous idea is suggested in [30] in a broader context of image understanding. Knowing when the model fails indicates that the assumptions about the scene are either wrong or insufficient, leading to the investigation of alternative models in a backtracking procedure.

In these methods, detection and recovery from failure is implemented with an automatic procedure based on complex reasoning, without guarantee that the correct result will be obtained under all circumstances. To account for vague, uncertain, incomplete and inconsistent medical knowledge that is difficult to represent explicitly [27], in FIS the evaluation of results is done by the user based on visual feedback on the screen, and the detection of an error triggers an interactive correction procedure. In our approach, interaction and computation are integrated in one process (requirement #1), guaranteeing reliable and clinically acceptable results.

**The user can help to prevent failure.**

A review in [15] illustrates how expert systems are used to build a sequence of operations for image segmentation on the basis of high-level goals specified by a
human operator. The underlying assumption is that the user can serve as a source of complementary knowledge to prevent errors and assist in complex tasks [9]. In general, user participation is expected in two situations: to specify the high-level goal of segmentation and confirm or reject the result obtained by the method. A failure detected by the user triggers a backtracking process based on fixed rules and a new result is presented on the screen. This trial-and-error strategy can lead to a tiring process when the reasoning process cannot handle the problem at hand. The system described in [9] reduces this problem by asking the user an “example” of the expected result and other “hints” providing contextual knowledge. This information is used to configure the segmentation process and the parameters for the low-level image processing operators.

In the cases above, the user sees the segmentation process as a “black box,” with the advantage of intuitive operation. However, the information provided by the user is limited to the segmentation target, with typically low semantic value. In FIS, we assume that, when the adequate user interface is available, the user can provide more specific information at a higher semantic level (requirement #4), facilitating the reconfiguration of the computational method.

The user can help to recover from fail.

In [27] the idea of looking at the limits of current computer vision methods (i.e. when and why they fail) is put forward as a way to determine the necessary level of user interaction. This idea is complemented in [25] and [22], where artificial intelligence components are suggested as a way to limit user interaction to the situations where the system really needs an expert’s solution. No objective approach to the issue is proposed.

Oracle systems complete this gap by proposing a strategy where “information input by the user results in a reasoning process in which both the user and the machine are involved in turns, and that results in a change in the machine’s representation of data” [28]. The picture on the screen shows a visualisation of results obtained by the computer, which is used to mediate the exchange of information between the user and the machine. The user, who has the knowledge of the application context, visually inspects these results and criticises them using a textual interface. The computer uses this critique to revise the data, producing new results, etc. The types of criticisms are limited and correspond to different problems that can be detected by the user based on the visual information.

Although oracle systems were proposed with a different motivation, the underlying conceptual model is similar to FIS. The major improvement of our method consists of a more intuitive and effective user interface, where corrections are indicated via graphical tools with high semantic level (requirement #4), as opposed to the textual communication language adopted in [28].
3.5 Conclusions

In the first place, by objectively defining the conditions for the generation of reliable, accurate and reproducible segmentation results with efficient interaction, the requirements for interactive segmentation formulated here represent a step toward guidelines for the design of interactive segmentation methods.

Secondly, the framework presented here constitutes a valuable conceptual model for the design of interactive segmentation methods, in contrast with the usually *ad-hoc* approach to interaction adopted so far. Our strategy introduces the notion of *case-based analysis of failure* of the computational method as an instrument to define how interaction is added to the segmentation process. From this analysis, a limited number of cases of failure are discriminated and the adequate recovery procedure for each one is identified. Structuring user intervention into cases facilitates the design of the user interface, with efficient interactive tools and effective visual feedback. Moreover, it provides a straightforward basis to reason upon user input to steer computation, paving the way for the development of interactive tools with truly intelligent behaviour. The structured approach proposed here leads to the design of interactive segmentation methods that address all the aforementioned requirements. Unlike other methods described in the literature, methods based on FIS guarantee the efficient generation of reliable, accurate, and reproducible segmentation results.

The implementation of a method based on this framework, however, poses high demands on the supporting computational method. As the first demand, FIS makes sense only when the underlying computational model is sophisticated. Note, for example, the difference in the corrections for case #1 (low visual evidence) in the paintbrush and deformable model methods. Since the first method does not adopt shape features in the segmentation model, the correction procedure must resort exclusively on the curve drawn by the user, possibly leading to less reproducible results than the second method, where the smoothness constraint is also taken into account. As the second demand, FIS requires subtle control of parameters and their impact on the segmentation result. This issue turned out to be less trivial than expected for the methods presented here, indicating that straightforward implementation of FIS as an add-on to existing methods is difficult to achieve.

In spite of that, the examples in section 3.3 show that FIS, seen as a theoretical model, holds when put against two existing paradigms for interactive segmentation, namely deformable models and the active paintbrush. For the methods at hand, the analysis of situations for failure appeared to be simple when guided by the general cases of the FIS model. Likewise, the design of efficient interactive tools and effective visual feedback appeared simpler when cases of failure and the necessary information for correction are determined according to this model. The framework has proven to be useful also in the design and development of an interactive segmentation solution for a difficult clinical application described in chapter 6, which could not have been solved by an automatic method.
Thanks to the Image Sciences Institute, University Hospital Utrecht, for providing the images used in this study, and to Marcos Cordeiro D'Ornellas, Erik Engbers and Tat Hieu Nguyen, for their help and suggestions.

Bibliography


Chapter 3. A Framework for Interactive Segmentation


