Received: 13 October 2016 | Revised: 5 January 2017 | Accepted: 5 January 2017

DOI 10.1002/rcs.1813

WILEY The International Journal of Medical Robotics and Computer Assisted Surgery

ORIGINAL ARTICLE

# Generative Anatomy Modeling Language (GAML)

Doga Demirel[1] | Alexander Yu[2] | Seth Baer-Cooper[2] | Tansel Halic[2] | Coskun Bayrak[1]

[1] Department of Computer Science, University of Arkansas at Little Rock, USA

[2] Department of Computer Science, University of Central Arkansas, USA

**Correspondence**
Tansel Halic, Department of Computer Science, University of Central Arkansas, USA.
Email: tanselh@uca.edu

**Funding information**
National Institute of General Medical Sciences, Grant/Award Number: P20 GM103429; National Institutes of Health, Grant/Award Number: NHLBI 1R01HL119248-01A1, NIH/NIBIB 2R01EB005807, 5R01EB010037, 1R01EB009362 and 1R01EB014305.

## Abstract

**Background** This paper presents the Generative Anatomy Modeling Language (GAML) for generating variation of 3D virtual human anatomy in real-time. This framework provides a set of operators for modification of a reference base 3D anatomy. The perturbation of the 3D models is satisfied with nonlinear geometry constraints to create an authentic human anatomy.

**Methods** GAML was used to create 3D difficult anatomical scenarios for virtual simulation of airway management techniques such as Endotracheal Intubation (ETI) and Cricothyroidotomy (CCT). Difficult scenarios for each technique were defined and the model variations procedurally created with GAML.

**Conclusion** This study presents details of the GAML design, set of operators, types of constraints. Cases of CCT and ETI difficulty were generated and confirmed by expert surgeons. Execution performance pertaining to an increasing complexity of constraints using nonlinear programming was in real-time execution.

KEYWORDS

airway management, authentic human anatomy, modeling language for human anatomy, nonlinear constraints framework, nonlinear programming, virtual human anatomy

## 1 | INTRODUCTION

This paper describes the Generative Anatomy Modeling Language (GAML) framework, which allows modification of 3D base anatomy models using human readable and simple commands in real-time. The process of design, creation, and refinement of a 3D model is an extensive and laborious task, which is inevitable in the development of virtual simulators. The process involves in-depth analysis to determine the necessary models in collaboration with expert physicians. This then requires numerous follow-up meetings with 3D designers and the engineering team. The 3D designers, having expertise in using 3D design tools with minimal to no knowledge in physiology and human anatomy, generate the 3D geometry and textures (e.g. images over the 3D models) with input from the engineering team. The overall objective is to reflect the expert physicians' feedback for the area of the interest. The final output from this design phase is often not useful and requires additional modifications and refinement (e.g. alteration of geometry, textures, polygon decimation, removal of unseen parts, etc.). The refinement process is necessary to make the models robust for the physics simulation and fast for visual rendering, however, the refinement process can result in anatomically incorrect models. Therefore, the design–feedback–redesign process for each 3D model requires many iterations over time to have anatomically correct, realistic and acceptable model for real-time physics simulation and visualization.

Virtual surgery simulators often support various 3D scenarios that enable physicians to practice with difficult or not common cases. These cases stem from the variations and aberrations of the anatomy. The scenarios for each of these cases experience a similar entire design iteration process even when there exists a 3D base scenario. However, creation of difficult scenarios by altering a base 3D model is challenging due to the complexity of human anatomy, which often requires hand-tailoring of models in professional software packages. The entire process of variation generation from base models can become as arduous a task as creating a 3D base scenario from scratch.

## 2 | BACKGROUND

The literature for generation of the models and modeling languages extends over multiple disciplines with a focus on variety of domain-

  

specific problems. Common anatomy modeling language (CAML)[1] offers a single framework that allows developers to create human anatomy models by starting with generic building blocks (human modeling primitives (HMPs)), then specializing to each model's need. HMPs are connected, as cells in the body, to form an organ, and organs are connected to form organ systems. While CAML can be deployed in a collaborative environment for medical simulator development team due to the common syntax of the language, it does not allow editing models in real-time with realistic constraints.

Several languages,[2–4] are developed for modeling and animating 3D models. HyperFun[2] is a web based high-level programming language aimed at modeling a diverse range of 3D objects such as fractals, human anatomy, and geological structures using implicit surfaces. In the work by Morkel and Bangay et al.,[3] techniques are outlined to improve procedural modeling using a set of operations such as selection, curve-shaping, and extrusion etc. These techniques are not defined as a new language, but are used as a tool to improve 3D models while avoiding as much manual intervention as possible. Cutler et al.[4] describes a simple scripting language that generates and modifies complex volumes with simple input meshes. While these languages are efficient enough for object alteration, muscle and bone animation, they are yet to be efficient enough for human model modification and skin deformation with respecting the anatomical features. Studies in[5,6] are focused on optimization in 3D models. Lohikoski et al.[5] focused on mesh optimization in order to see performance improvements in their 3D scene. The main optimization performed was poly-count reduction. During poly-count reduction, the meshes were simplified but not to the point of losing aesthetic appeal or familiarity. These optimizations yielded a 40% increase in frame rate and a 42% decrease in memory usage. In Pighin et al.[6] 3D, models were constructed from a video feed and were to be used in animation. They used a continuous optimization technique involving a stochastic gradient that estimates second-order derivatives by sampling a small number of pixels, while accounting for an error function.

There exist studies using medical imaging scans for constructing and then modifying 3D anatomy models. Kim et al.[7] used the marching cube algorithm on a 2D preprocessed image and then reconstructed the image into a Virtual Reality Modeling Language (VRML)[8] format. The models can then be adjusted by the length parameters in order to make variations. Kaus et al.[9] describes a fully-automated image-to-model technique for myocardium segmentation involving statistical point distribution models and surface meshes by the use of sampling multiple images. Although these techniques provide realistic models, the processes of construction of the models are long and tedious due to creation and cleaning of the constructed models. The focus of the work in Sierra et al.[10] outlines several techniques to generate common pathologies for surgical simulators with the use of cellular automation, skeleton based design and particle systems. The work in[11] uses a mesh generative approach on 3D shapes by using shape descriptions (e.g. a few specific parameters that characterize the shape, such as the polygon, offset vector, or even functions). These shape descriptions are used to create complex geometry from simple input geometry. This approach allows users to change the form of the model with high-level shape Euler operators, such as creating or deleting edges and splitting or merging faces. While this approach is useful for changing the shape object models, it does not allow integration of context information such as human anatomy needing a specific set of geometry constraints.

In this work, we proposed the GAML framework, which provides an easy to use platform for manipulation of 3D models, satisfying any geometric constraints imposed by the human anatomy. The formal foundation of GAML is defined in terms of context-free grammar. The language supports incorporation of the constraints in a 3D model (e.g. organ) varying with its location and types such as muscle, bone, joint, etc. We used a nonlinear optimization model where constraints can be dynamically added and removed.

Even though GAML can be used for constructing arbitrary 3D models specific to any region in the anatomy, it was deployed to generate difficult scenarios for the two critical airway management

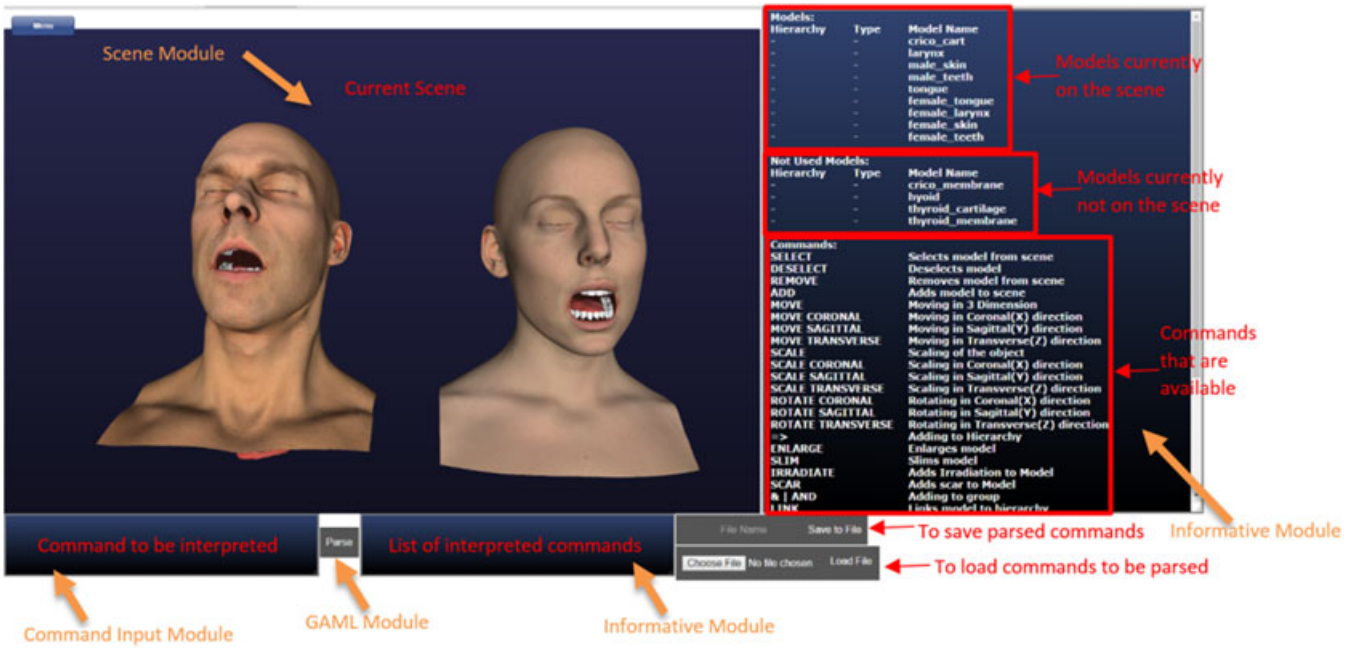


**FIGURE 1** GAML GUI

techniques; endotracheal intubation (ETI) and cricothyroidotomy (CCT). These difficult scenarios are a part of our ongoing work on development of the virtual airway skills trainer simulator,[12] and are representative cases of how GAML can be used.

## 3 | MATERIALS AND METHODS

### 3.1 | GAML architecture

We developed the GAML platform, as shown in Figure 1, using our Π-SoFMIS framework.[13,14] Π-SoFMIS is designed to build real-time interactive simulation and visualization applications on the web. The framework utilizes WebGL technology for realistic rendering and uses HTML5 elements for the user interface. Having the framework running on the web browsers allows hardware-independent, real-time, portable, and accessible 3D interactive multimodal applications to be built. In the framework, realistic rendering uses built-in shaders supporting various rendering capabilities such as shadow mapping, subsurface scattering, various materials, depth of field, etc. JavaScript Object Notation (JSON) for 3D models, generated from common file formats such as .OBJ and .3DS, are used to import geometry into the virtual scene.[15]

GAML platform has a front end graphical user interface (GUI), where commands from the console based on HTML elements can be given. Once the commands are received, based on the type of command, the GAML interpreter either modifies geometry directly (e.g. unconstrained movement) or computes solution to our nonlinear model first and then forwards the command to the GAML command executor stage. If the command is subject to constraints, the hierarchy of constraints is traversed and optimal solution of nonlinear model is computed. The solution is then checked against the boundary of the objects (size and location of one object against the objects in the scene). The final optimized solution is then used to update the original command with new parameters (e.g. optimal position). This update is then stored in the informative box display (see Figure 2). If the given command violates the constraints or boundaries, that means there is no feasible solution in our model, the command is rejected with appropriate message to the user. The functionality of each of the component and sample execution flow is illustrated in Figure 2.

#### 3.1.1 | GAML platform

The GAML GUI is composed of four major components: (a) command input module, (b) GAML module, (c) scene module, and (d) informative box modules. In command input module, the commands are parsed for lexical analysis and then sent to the GAML module for execution. The history of commands, such as saving/loading, to/from a file, and autocompletion features were integrated to further improve user friendliness of the platform. Informative box module has the list of current imported models, list and hierarchy of the constraints in the scene, current updates of the used/unused 3D models. The scene module is in charge of executing the commands provided from the GAML interpreter module or optimization module without altering them. The output of the commands is visualized in real-time.

#### 3.1.2 | GAML

GAML structure is defined based on context-free grammar. The language allows execution of commands that are based on simple and easy to understand wording. These commands are categorized by two types: first type of command is for direct 3D geometry modification (e.g. move, scale, scar generation etc.), while the second type (e.g. selection, joint construction, grouping etc.) is for several

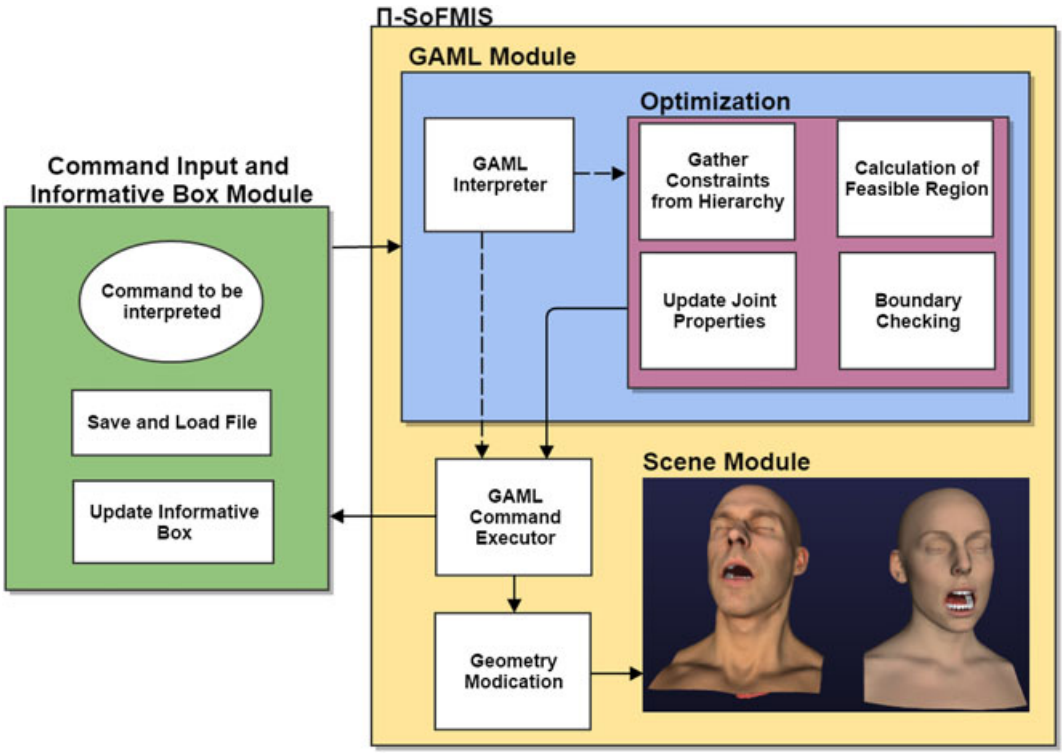


**FIGURE 2** Architecture of GAML: dotted lines show optional operations, and solid lines show the order of operation

hierarchy constructions (e.g. joints, links) and incorporation of geometry constraints. Commands involving modification of geometry (e.g. affine transformation or deformation of a model) comply with the defined hierarchy. The compliance is automatically handled with our nonlinear programming framework.

GAML context-free grammar is parsed using Jison , a JavaScript Bison[16] parser. Jison supports languages defined by, left-to-right, Look-Ahead left-to-right[17] (LALR) (k = 1), Left-to-right (LR) (k = 0), and simple left-to-right (SLR) (k = 1) grammars. Numbers in parenthesis represent the *k* value which specifies the number of tokens to look ahead during the parsing process. Parsing the context-free grammar can be divided in three phases; (a) Lexical grammar definition, (b) operator associations and precedence, and (c) definitions of rules.

Lexical grammar defines the tokenization rules in the language. These rules are pre-defined and used by the grammar parser. Defined rules need to comply with the operator precedence and associations. In lexical analysis, the tokenization phase decomposes command strings to generate words (called tokens) identifiable by the language. After the tokenization process, GAML processes the tokens to ensure that they are valid statements by following the lexical rules and the operators. The operator associations identify the operators as left or right operand and the operator precedence determines the order of execution. The lexical grammar, operator associations and precedence can be seen in Appendix A. Table 1 represents the context-free grammar and the tokens created for GAML for a 'select' command.

GAML supports both the low level (translate, scale, rotate, etc.) and the high level geometry modification commands (irradiation and scarring, adjust, etc.). These functions can either be performed manually on the location of the model by selecting stylus (e.g. sphere stylus as seen in Figure 3) or executing the irradiation and scarring commands using anatomical axes such as sagittal, frontal, traversal planes and coordinates. High level commands such as irradiation and scarring example output can be seen in Figures 3 and 4, respectively.

For all GAML commands, the anatomical plane can be used for the spatial positioning. The language also enables the use of operators such as '=>' and '+'. The '=>' operator assigns the selected hierarchy number to a model. If there are multiple models in the same hierarchy, any relative change (e.g. affine transformation, linked models) to a model affects the rest of the models in the hierarchy. If there exist multiple objects in the hierarchy, geometrical modification can be performed with respect to each other's auto computed axes aligned bounding box centers. The available commands and their functionalities have been listed in Tables 2, 3 and 4.

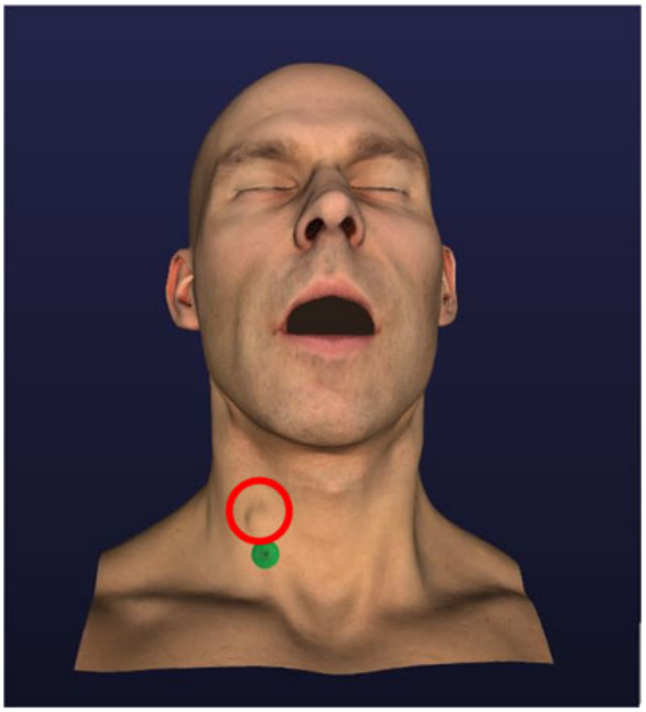

**FIGURE 3** Irradiation creation

Dimension parameters are defined to eliminate the need to put exact quantity in the commands such as 'a lot', 'more', 'much more', etc. (see Appendix A). These values can be used as a measure of movement, rotation or a parameter to a 3D model modifier operation (e.g. scar). The exact values, although they can be changed and depend on the operator, are proportional to the model size.

## 3.2 | Optimization model

Once hierarchies and the constraints are given for GAML, the 3D model can be manipulated. The modification could be as simple as a basic translation to a specified location, or it could be a command that

**TABLE 1** Pseudocode for functionality 'select'

```
// Definition of lexical grammar
'SELECT'|'select'              return "SELECT"
// Definition of precedence of order of operation
left SELECT
// Definition of language expression and functionality
SELECT model_selected
      GAML_Select_Command($Model)
// JavaScript function bound to the GAML
select command
   .
```

```
Token
Case 22: // Generated case number for the select
command token
 begin
   for every model in scene
      hide(model)
   enable(model_selected)
   for every model in scene
      if model.is Connected To
(model_selected)
            enable (model)
    EnableSelectionTool()
    Moller_Trumbore ()
    Draw_sphere (intersection points)
end
Return 31 // Generated case number for the
model token
/^(?:!SELECT|SELECT|select\b)/
// Generated rule for the use of select command
```


**FIGURE 4** Scar creation

**TABLE 2** Selection and identification commands

| Selection and identification command | Functionality |
|---|---|
| Select | Selects the model, and hides the other models. Exposes a wireframe sphere for local modifications. The sphere is projected on to the selected models by using the Moller-Trumbore ray/ triangle intersection algorithm.[24] |
| Deselect | Exposes the other models in the scene |
| Identify as | Identifies the selected area as a new object |
| Selection size | Changes the size and intensity of the wireframe sphere |
| AND/+ | Allows the user control multiple objects |
| Place angle joint | Allows the user to place an angle joint at a location |
| Place flexibility joint | Allows the user to place a flexibility joint at a location |
| Place absolute distance joint | Allows the user to place an absolute distance joint at a location |
| Place all joints | Allows the user to place a joint comprised by all joints at a location |
| Link with | Connects two objects that have a joint within a threshold |
| => | Appoints a hierarchy to the model |
| Add | Adds a new model to the scene |
| Remove | Removes a model from the scene |

results in 3D topology change. However, considering the given constraints, the specified location may not always be a feasible solution. As an example, when a move command is executed, the GAML solves our optimization model by seeking the optimum solution. This is achieved by using a solver featuring constrained optimization by linear approximation: an implementation of Powell's nonlinear derivative-free constrained optimization that uses a linear approximation approach (COBYLA)[18,19] Since the user enters a desired movement, the optimization objective function aims at a point in the feasible region, which is closest to the desired point. Our optimization model and the constraints used in GAML are given in Table 5.

In our model, $p_i$ is a new spatial 3D position or can be a constant current position of a joint $J_i$, where $J_i$ can be a node attached or a node in a 3D Mesh ($M$). $A$ is the set of joint pairs $(i,j)$ with absolute distance constraints, $B$ is the set of joint pairs $(i,j)$ with angle constraints, $B'$ denotes the set of 3-tuple elements $(i,j,k)$ such that joint $i$ is not allowed to pivot about joint $j$ and around the $k$-axis and $C$ is the set of joint pairs $(i,j)$ with flexibility constraints. Joint is an abstract definition in GAML that holds all constraints and attachment information. $N$ is the number of joints that can be dynamically added and removed in the objective function. For every joint, there can exist a movement within the user defined constrained space. For each pair of joints $i$ $(p_i)$ and $j$ $(p_j)$, there can be up to four different constraints given in Equations (1)–[4] (see Table 5) that limit the movements. $\Theta_{ij}$ is the maximum angle that joint $i$ $(p_i)$ is allowed to pivot about joint $j$ $(p_j)$. $p_{io}$ is the initial point for joint $i$. The axis of rotation, if desired, can be locked with equation 2a. $(\boldsymbol{p_{io}} - \boldsymbol{p_j})_{k\,-\,axis}$ and $(\boldsymbol{p_i} - \boldsymbol{p_j})_{k\,-\,axis}$ are the $k$-axis components of the corresponding directional vectors, where $k \in \{x, y, z\}$. $(\boldsymbol{p_{io}} - \boldsymbol{p_j})$ is the direction vector between the original position of joint $i$ $(\boldsymbol{p_{io}})$ and the pivot point of the joint $j$ $(p_j)$, thus making the new position of point $i$ $(p_i)$ the

**TABLE 3** Geometrical modification commands

| Geometrical modification command | Functionality |
|---|---|
| Scar | Creates a scar at the location of sphere |
| Scar Size | Changes the size of the scar |
| Irradiate | Creates an irradiation at the location of sphere |
| Irradiation Size | Changes the size of irradiation |
| Create Tear | Creates a tear at the location of sphere |
| Tear Size | Changes the size of the tear |
| Enlarge | Enlarges the selected region |
| Enlarge Until | Enlarges the selected region with the given step size |
| Adjust | Adjusts the vertices of the model selected by the sphere |
| Adjust overall | Adjusts the vertices of the selected model |
| Smooth | Smooths the selected model |
| Tessellation | Tessellates the selected model |
| Displace outward | Create displacement outward on the selected model |
| Displace inward | Create displacement inward on the selected model |
| Duplicate | Duplicates the selected model |

**TABLE 4** Movement commands in GAML

| Movement command | Functionality |
|---|---|
| Move Coronal | Moves the selected model in the coronal (X) axis |
| Move Sagittal | Moves the selected model in the sagittal (Y) axis |
| Move Transverse | Moves the selected model in the transverse (Z) axis |
| Scale Coronal | Scales the selected model in the coronal (X) axis |
| Scale Sagittal | Scales the selected model in the sagittal (Y) axis |
| Scale Transverse | Scales the selected model in the transverse (Z) axis |
| Rotate Coronal | Rotates the selected model in the coronal (X) axis |
| Rotate Sagittal | Rotates the selected model in the sagittal (Y) axis |
| Rotate Transverse | Rotates the selected model in the transverse (Z) axis |

**TABLE 5** Nonlinear optimization model and constraints for transformation of a node

| | | |
|---|---|---|
| ***Min:*** $\sum_{I=1}^{N} k_I \lvert \boldsymbol{p}_I - \boldsymbol{p}_{Destination} \rvert$ | | |
| *Subject to:* | | |
| $Dist_{ij} - \lvert \boldsymbol{p}_i - \boldsymbol{p}_j \rvert = 0,$ | for $(i,j) \in A$ | *(1)* |
| $cos^{-1}\left(\frac{(\boldsymbol{p}_{io}-\boldsymbol{p}_j)\cdot(\boldsymbol{p}_i-\boldsymbol{p}_j)}{\lVert(\boldsymbol{p}_{io}-\boldsymbol{p}_j)\rVert\times\lVert(\boldsymbol{p}_i-\boldsymbol{p}_j)\rVert}\right)-\theta_{ij}<0,$ | for $(i,j) \in B$ | *(2)* |
| $(\boldsymbol{p}_{io} - \boldsymbol{p}_j)_{k\,-\,axis} - (\boldsymbol{p}_i - \boldsymbol{p}_j)_{k\,-\,axis} = 0,$ | for $(i,j,k) \in B'$ | *(2a)* |
| $Dist_{ij}-\Delta d_{\max}-\lvert \boldsymbol{p}_i-\boldsymbol{p}_j \rvert<0,$<br>$\lvert \boldsymbol{p}_i-\boldsymbol{p}_j \rvert-Dist_{ij}-\Delta d_{\max}<0,$ | for $(i,j) \in. C$ | *(3)*<br>*(4)* |
| $i, j \in J$ *and* $i, j \subseteq M, k_I > 0, A \cap C = \varnothing$ | | |

decision variable. $(\boldsymbol{p}_i - \boldsymbol{p}_j)$ is the direction vector between the new computed optimal position of the joint $i$ $(p_i)$ and the pivot point of joint $j$ $(p_j)$. $Dist_{ij}$ is the original distance between two joints $i$ $(p_i)$ and $j$ $(p_j)$ and $\Delta d_{max}$ is the maximum displacement allowed between the joints $i$ $(p_i)$ and $j$ $(p_j)$ and is calculated using the stiffness ratio $k_I$. Equation (1) satisfies the absolute distance constraint, Equation (2) satisfies the angle constraint, and Equations (3)–(4) satisfy the flexibility constraint. Details of the constraints are further explained in the next section.

## 3.3 | Constraints

Our constraints are classified with respect to its geometry association constraints (GAC) which can be distance, angle, connectivity, etc. GAC defines how to attach GAML joints on an object. Once a joint is attached, the object motion will be constrained in the joint's permissible movement. GAC can link two objects together within the spatial distance threshold, which is defined by the user. If no joint is located in the vicinity of object A within a threshold distance, a new joint is created at the closest point of the Object A to the Object B. The two closest joints will hold the information about each object and the list of constraint equations about the respective joints attached to their objects. Once the joints are placed on the model, joints can be linked together with the GAML command 'link with'. The 'link with' command connects two models that have joints within a specified threshold. Any geometric modification on an object will propagate a corresponding action performed on the rest of its connected objects. For every 3D object in the scene, a type can be assigned. Available types in GAML are: muscle, joint, vein, artery, bone, ligament, fat, teeth, skin, and cartilage. Given the assigned type, the object is declared as either a rigid or a deformable object. Deformable objects can be muscle, vein, artery, ligament, fat, skin, etc., while rigid objects can be bone, teeth, and cartilage. Rigid objects will move relative to its attached joints. Deformable objects can be stretched via a linear skinning formula based on the distance formula in the case of a motion. Figure 5 shows the joint representation of a model with two allowable joint connections.

Once the joints are linked, additional constraints can be coalesced on the joints by linking them with other joints so that they can model specific articulation. GAC are classified in Type-I and Type-II constraints.

**Type I Constraints.** These constraints are automatically generated when models are imported in the virtual scene. Currently, the Type-I constraints include location and size. The location constraint limits the transformation of a model in the scene. For instance, when there is an attempt to move a model, the location constraint prevents penetration into other objects by constraining the motion. Similarly, bounding limits of the model are extracted from the geometry of the anatomical model and is also constrained by other objects around it. If the user attempts to increase the size of the object, the object's size will not allow protruding into another object. The user is allowed to specify a maximum distance on the location constraint that limits the distance that an object

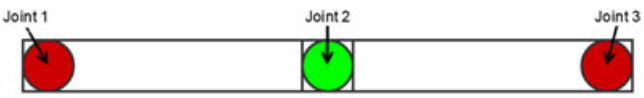


**FIGURE 5** Representation of joint structure with two allowable joints connections (red circles) and green circles indicate linked joints to the model

can move. The user can also set a minimum or maximum on the size constraint, this will ensure that the object's size cannot exceed a preset size.

**Type II Constraints.** Unlike Type-I constraints, Type-II constraints require explicit declaration. These constraints can be relative distance, angle, and geometric flexibility (e.g. allowable stretch ability). Type-II angle constraint (Figure 6) allows for joint motion within the preset angle with respect to another joint as defined in Equation (2) in Table 5.

Rotation of joints can be constrained to one axis. To fix the rotation to one axis, the previous directional vector $u$ and the current directional vector $v$ should have the same direction in the axis as given in Equation (2a). Any of the axes can be locked and freed with this formulation.

Absolute distance (Figure 7), refers to the exact distance between two joints that is maintained at all times. This constraint is generated when two joints are declared to keep a relative distance or the joints are attached to the same non-deformable object as given in Equation (1) in Table 5.

Flexibility constraint (Figure 8) is used for the cases where deformation is allowed between two joints. This constraint varies depending on the type of the anatomy, which affects the stiffness ratio ($k$) in the objective function (e.g. a bone has a higher stiffness than a muscle). Based on the stiffness ratio, a maximum distance $\Delta d_{max}$ is calculated and used in Equations (3)–[4] in Table 5 (see Figure 8 for representation) to constrain the maximum and minimum distances. If a joint update requires deformation, linear skinning algorithm is applied to the model. This flexibility constraint can be only generated if the object is deformable.

A virtual 3D scene of the laryngeal anatomy and representations of constraint joints in the scene are presented in Figure 9. Green circles represent the joints on the object, while red circles represent the joints that are in the user defined threshold. In an attempt to modify the larynx position to a desired location (without optimization model after transformations), as shown in Figure 10, our optimization model computes the optimum location (see Figure 11) that avoids the creation of irrational anatomy. The larynx is translated to an optimum location and the constraints (relative distance, angle, flexibility, connectivity) for the larynx and skin are satisfied. The representation of the scene after the translation and the modified areas are marked with red circles is given in Figure 11.

## 4 | RESULTS

We have measured the execution time to solve for the solution to our nonlinear optimization model with different numbers of constraints and joints. In order to benchmark execution performance for varying constraints on one joint, randomly generated constraints were added

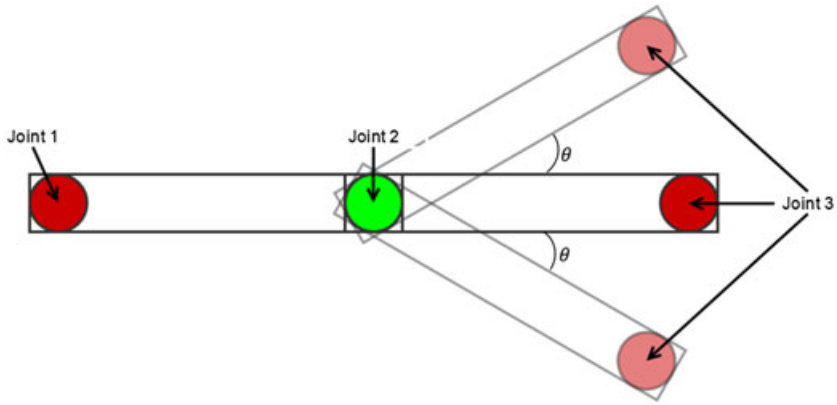


**FIGURE 6** Angle constraint on a joint. Transparent boxes indicate the maximum location that the object can move

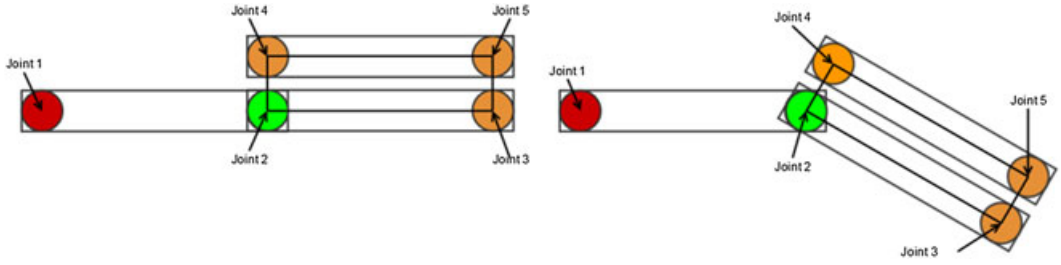


**FIGURE 7** Joints that are constrained by absolute distance. Orange joints are constrained by absolute distance. Green joints are constrained by angle and absolute distance

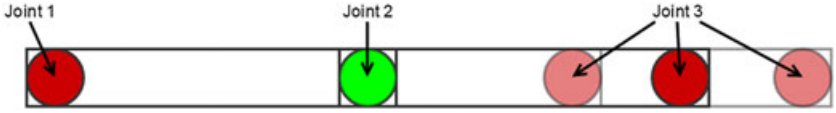


**FIGURE 8** Flexibility constraint on joint. Transparent boxes indicate the minimum and maximum amount that the object can deform

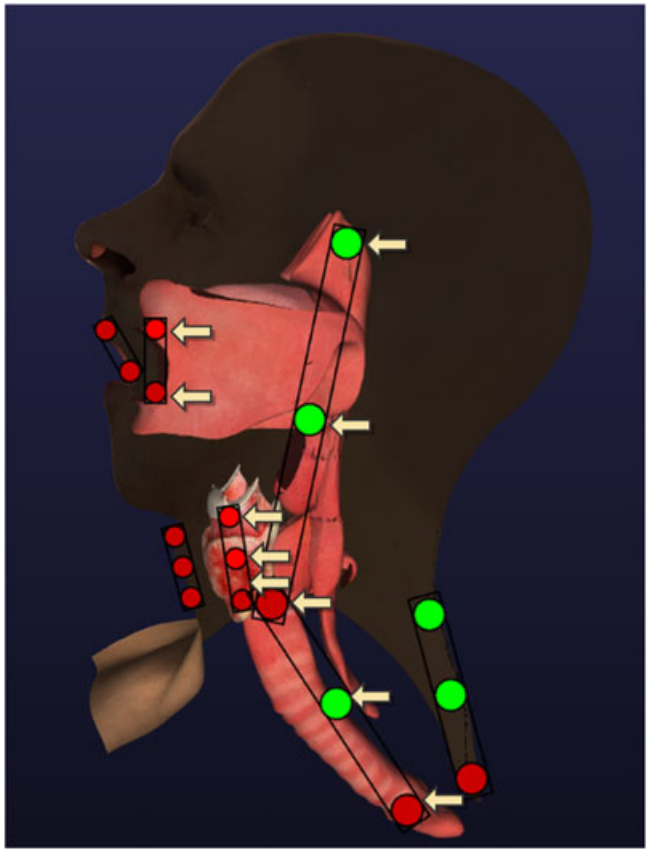

**FIGURE 9** Joint representation of the models before transformation

**FIGURE 10** Transformation without optimization model

to the joint. Figure 12 indicates an increasing number (up to 100) of flexibility constraints, angle constraints, distance constraints, and all (flexibility, angle, relative distance) constraints on one joint and the

**FIGURE 11** Transformation with optimization model with constraints

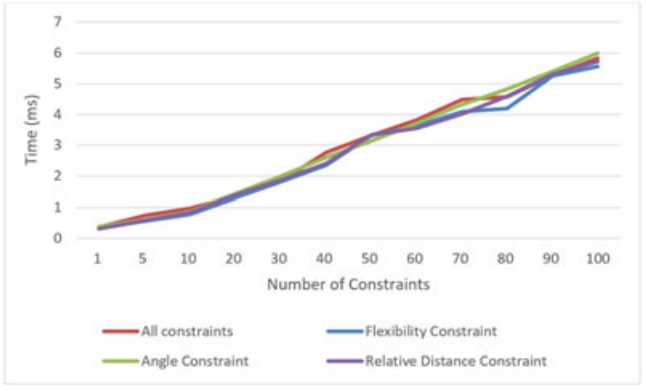


**FIGURE 12** Execution performance with increasing number of constraints

time required for the solver. We also tested increasing number of joints. Figure 13 indicates an increasing number (up to 40) of joints subjected to an increasing randomly generated flexibility and angle constraints (40 joints =158 constraints). Each joint is also subjected to the absolute distance constraint with the previously added movable joint. The performance measurement was performed on an Intel Core i7-5820 K CPU, with 16.0 GB of memory and a GeForce GTX 970 (version 372.70) graphics card.

As the amount of constraints increase, the time required to solve the constraints increase exponentially. However, most scenarios should be well under the threshold (<1000 constraints) for the performance to scale relatively linearly. Similarly, as the number of joints increases exponentially and in most scenarios should be under the threshold (<20 joints) for the performance to scale relatively linearly. In both cases, the solution time is within the real-time frame rates for complex scene with hundreds of constraints and joints.

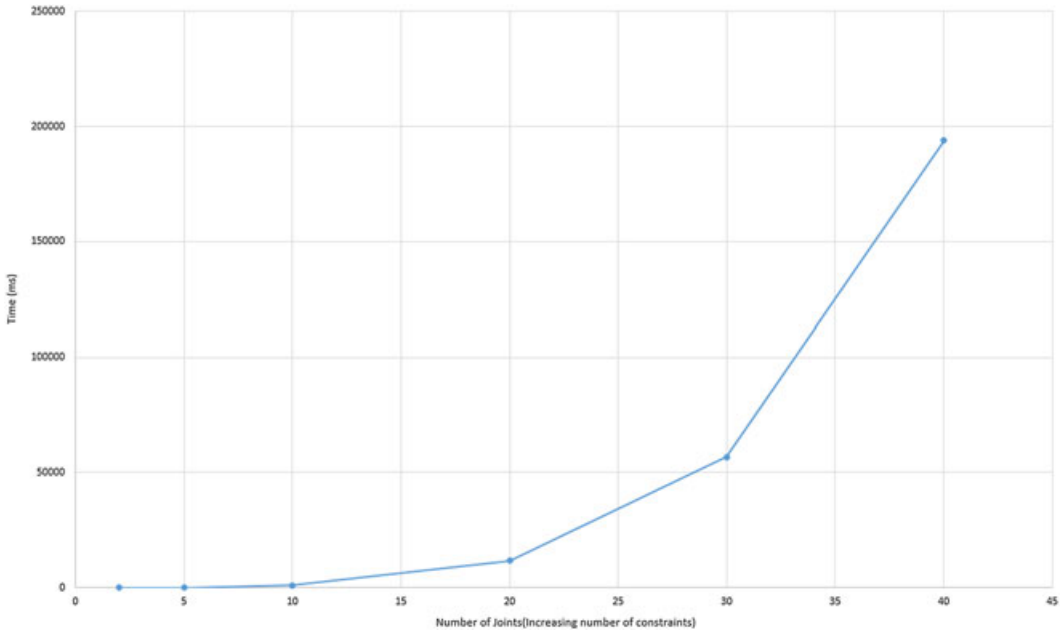


**FIGURE 13** Execution performance with increasing number of joints subject to increasing number of constraints

## 5 | DISCUSSION

GAML was used to generate difficulty scenarios for ETI and CCT procedures. These scenarios will be used in our virtual airway skills trainer (VAST),[20] which is a real-time surgical simulation platform aimed at training for ETI and CCT procedures for potentially challenging cases.

ETI and CCT are procedures used as a part of difficult airway algorithm (DAA)[21] to secure the airway of the patient. One of the most common techniques in intubation is ETI, whereas the CCT is the invasive substitute used in complicated cases as an alternative. In,[22] the ideal technique is chosen according to the patient's condition. A few seconds can be crucial in patients' health, selecting the optimum technique can prevent life threatening complications. Therefore, training with different difficulty cases is very critical.

Prior to the generation of the 3D models, we define the scenarios at different difficulty levels. The relevance of each scenario was confirmed by expert physicians. In these scenarios, each difficulty factor yields scores of 0, 1, or 2. In Tables 6–9, red colored difficulty factors are non-0 factors. The difficulty scenarios are defined in such a way that the higher the total score, the harder the intubation will become.

### 5.1 | Cricothyroidotomy difficulty scenarios

In CCT difficulty scenarios, eight airway difficulty factors were used. These airway factors were gender, body mass index (BMI), cricoid cartilage (CC) position, cricothyroid membrane (CM) dimension, neck extension (NE), neck length (NL), neck history (NH), and environment settings (ES). Difficulty level 1 (easy) is 0 to 3 points with none of these factors over 1 point. Difficulty level 2 (medium) is 4 to 7 points with at most one factors with 2 points. Difficulty level 3 (hard) is 8 or 9 points and difficulty level 4 (extremely hard) is 10+ points.

In the gender factor, males are assumed to be easier to intubate than females due to males having an Adam's apple. This anatomical landmark simplifies locating the thyroid cartilage, which is critical location to identify the cricoid membrane where the incision is performed.

**TABLE 6** Factor based scoring system for CCT

| Scenario | Gender | BMI | CC Position | CM Dimensions | NE | NL | NH | ES | Total Points |
|---|---|---|---|---|---|---|---|---|---|
| Base | Male | Normal | Regular | >10.4 mm | >45 | >1 | - | - | 0 |
| Easy-1 | Female | Normal | Low | >10.4 mm | >45 | >1 | - | - | 2 |
| Easy-2 | Male | Obese | Regular | >10.4 mm | >45 | <1 | - | - | 2 |
| Easy-3 | Male | Obese | Regular | >10.4 mm | 45 | >1 | - | LC | 3 |
| Medium-1 | Female | Obese | Regular | <10.4 mm | <45 | >1 | - | CP | 6 |
| Medium-2 | Male | Obese | Regular | <10.4 mm | <45 | <1 | NS | -- | 6 |
| Medium-3 | Female | Normal | Low | <10.4 mm | >45 | <1 | - | L | 5 |
| Hard-1 | Female | Obese | Regular | >10.4 mm | 45 | <1 | NI,NS | LC,CP | 8 |
| Hard-2 | Male | Obese | Low | <10.4 mm | <45 | <1 | NI,NS | - | 8 |
| Hard-3 | Female | Obese | Regular | <10.4 mm | 45 | <1 | NI | L,LC | 8 |
| Extremely Hard-1 | Female | Severely Obese | Low | >10.4 mm | <45 | <1 | NI,NS | LC,CP | 11 |
| Extremely Hard-2 | Female | Severely Obese | Low | <10.4 mm | <45 | <1 | NS | L,LC,CP | 13 |
| Extremely Hard-3 | Male | Severely Obese | Regular | <10.4 mm | <45 | <1 | NI/ NS | L,LC, CP | 10 |

**TABLE 7** Point based scoring system for CCT

| Scenario | Gender | BMI | CC Position | CM Dimensions | NE | NL | NH | ES | Total Points |
|---|---|---|---|---|---|---|---|---|---|
| Base | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Easy-1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 2 |
| Easy-2 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 2 |
| Easy-3 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 3 |
| Medium-1 | 1 | 1 | 0 | 1 | 2 | 0 | 0 | 1 | 6 |
| Medium-2 | 0 | 1 | 0 | 1 | 2 | 1 | 1 | 0 | 6 |
| Medium-3 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 5 |
| Hard-1 | 1 | 1 | 0 | 0 | 1 | 1 | 2 | 2 | 8 |
| Hard-2 | 0 | 1 | 1 | 1 | 2 | 1 | 2 | 0 | 8 |
| Hard-3 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 2 | 8 |
| Extremely Hard-1 | 1 | 2 | 1 | 0 | 2 | 1 | 2 | 2 | 11 |
| Extremely Hard-2 | 1 | 2 | 1 | 1 | 2 | 1 | 1 | 3 | 13 |
| Extremely Hard-3 | 0 | 2 | 0 | 1 | 2 | 1 | 1 | 3 | 10 |

**TABLE 8** Factor based scoring system for ETI

| Scenario | Mallampati Class | TD | HNM | BMI | Prominent incisors | Inter-incisor Gap | ULBT | Total Points |
|---|---|---|---|---|---|---|---|---|
| Base | Class 1 | >6.5 cm | >90 | Regular | No | >5 cm | Class 1 | 0 |
| Easy | Class 1 | 6.25 cm | 90 | Regular | No | >5 cm | Class 1 | 2 |
| Moderate | Class 2 | >6.5 cm | 90 | Regular | No | 4.75 cm | Class 3 | 5 |
| Hard | Class 3 | 5.5 cm | 90 | Obese | Yes (0.3 cm) | 4.2 cm | Class 3 | 10 |
| Extremely Hard | Class 4 | 5.5 cm | <90 | Severely Obese | Yes (0.6 cm) | 3.5 cm | Class 3 | 14 |

**TABLE 9** Point based scoring system for ETI

| Scenario | Mallampati class | TD | HNM | BMI | Prominent incisors | Inter-incisor gap | ULBT | Total points |
|---|---|---|---|---|---|---|---|---|
| Base | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Easy | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 2 |
| Moderate | 1 | 0 | 1 | 0 | 0 | 1 | 2 | 5 |
| Hard | 2 | 2 | 1 | 1 | 1 | 1 | 2 | 10 |
| Extremely Hard | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 14 |

CCT on a male is 0 points while on a female it is 1 point. The BMI of the patient affects the incision and makes it challenging to locate the landmarks and incise the skin. Therefore, a BMI rating of normal is 0 points, obese is 1 point and severely obese is 2 points.

CM dimensions are a difficulty factor due to the fact that most commercial kits use a 6 mm tube for the CCT procedure. CM dimensions over 10.4 mm high are 0 points, while CM dimensions less than 10.4 mm high are 1 point. Under the CM is the CC and the position of the CC affects difficulty; anatomy with regular CC position are 0 points, while low CC position are 1 point. Another factor is the NE. If the NE is over 45°, it is considered as 0 points. If NE is around 45 degrees, it is given 1 point and for the cases less than 45 degrees, it is 2 points. NL will affect the positioning of bones and tissues inside the anatomy. Equal or more than 1 cm distance between the inferior cricoid cartilage and the suprasternal notch is 0 points, less than 1 cm distance between the inferior cricoid cartilage and the suprasternal notch is 1 point. For the NH assessment factor, if there was no prior irradiation or neck surgery it is 0 points, if the patient had a prior neck irradiation (NI), tumor or prior neck surgery (NS) is 1 point and if the patient had both prior NI, tumor and prior NS it is 2 points. Optimum ES is the operating room or emergency room, which are 0 points. Light constraints (LC) are 1 point, Complicated positioning (CP) and Lack of tools (L) are 1 point each. Each of the suboptimal factors add 1 point. Table 6 shows the factor-based system while Table 7 shows the point-based scoring system.

The base case for CCT is shown in Figure 14. In Figure 15, the generated Extremely Hard-2 case is shown using our GAML platform. The fat level of the base model was changed to severely obese. Neck scar was added to the incision location. CM dimension was decreased to under 10.4 mm by reducing the size of the cricoid membrane, and the thyroid cartilage and cricoid has been moved closer to each other. Other factors such as neck extension are incorporated as a part of phsyics engine of our virtual simulator.

## 5.2 | Endotracheal intubation difficulty scenarios

In ETI difficulty scenarios, seven airway assessment factors were used.[23] These airway assessment factors were Mallampati

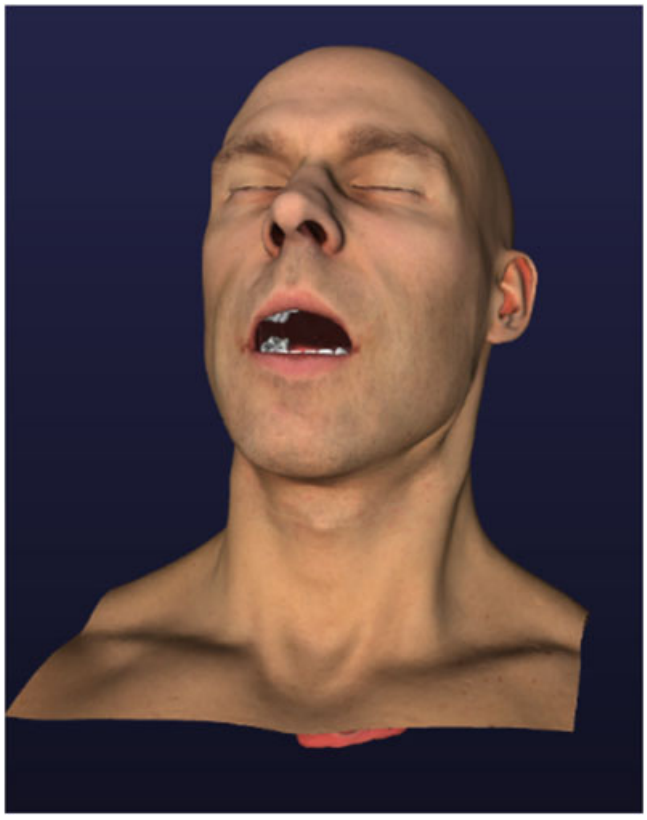

**FIGURE 14** Base case

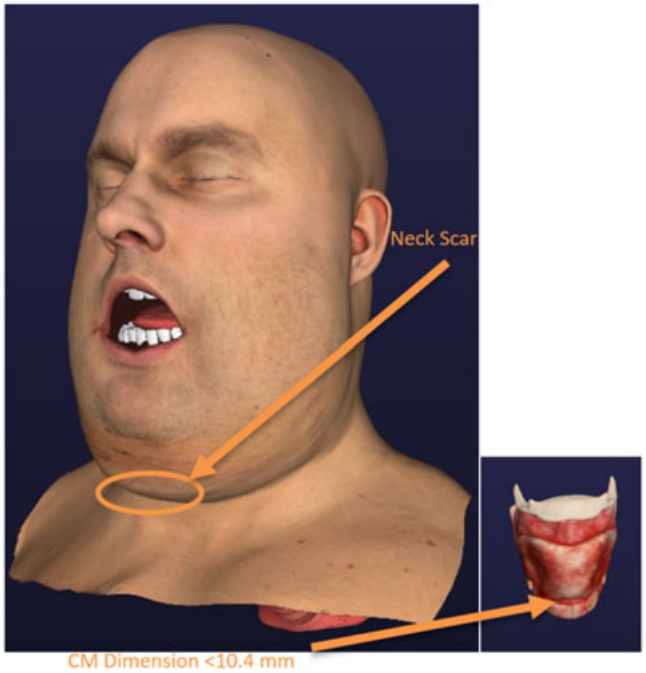


**FIGURE 15** Extremely hard-2 case created in GAML

classification, Thyromental Distance (TD), Head and Neck Movement (HNM), BMI, prominent incisors, inter-incisor gap, and Upper Lip Bite Test (ULBT). Difficulty level 1 (easy) varies from 0 to 2 points, with none of the assessment factors over 1 point. Difficulty level 2 (moderate) varies from 3 to 6 points, with at most one assessment factor with

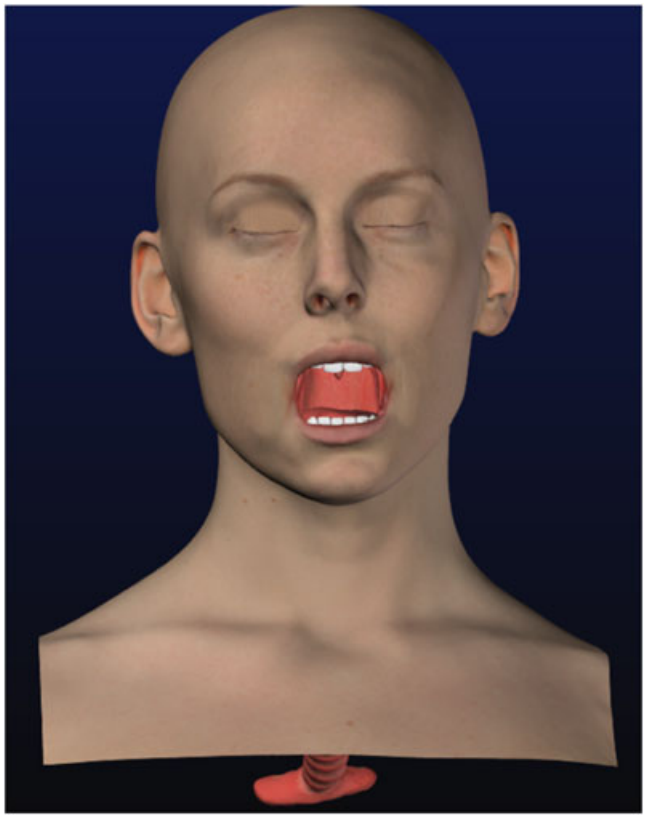

**FIGURE 16** Base case

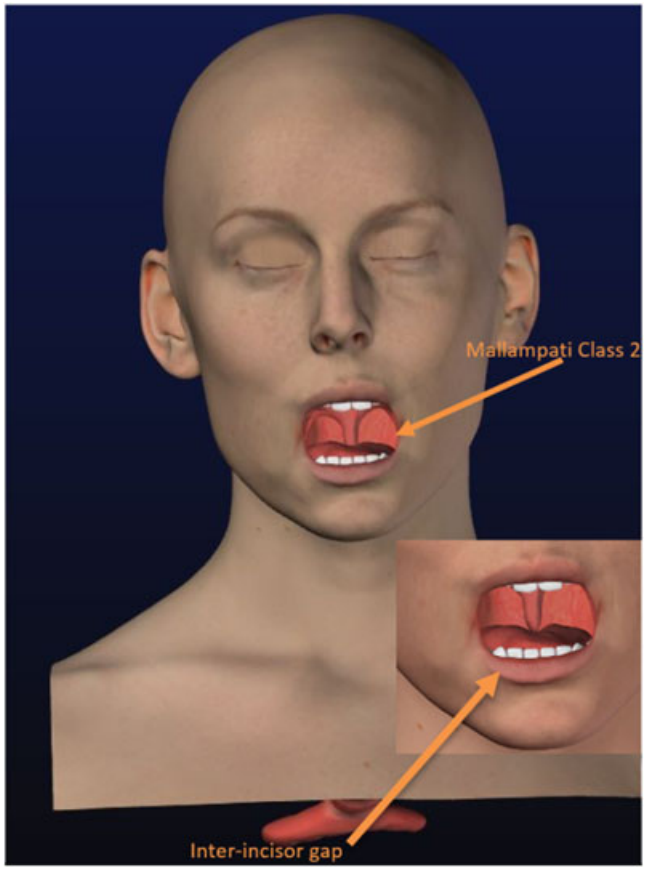


**FIGURE 17** Moderate case created in GAML

2 points. Difficulty level 3 (hard) scores are between 7 and 11 points and difficulty level 4 (extremely hard) is when the total difficulty score is equal or more than 12 points.

The first assessment factor is the Mallampati classification. Mallampati Class-2 features a bigger uvula, while Class-3 and Class-4 feature a bigger tongue size and large soft tissue. Mallampati Class-1 is the base case, which is 0 points. Mallampati Class-2 is 1 point, while Class-3 and Class-4 are 2 points. TD is the distance between the tip of the jaw to the thyroid notch. A TD of more than 6.5 is 0 points, 6 to 6.5 cm is 1 point, and less than 6 cm is 2 points. HNM is an assessment factor for the sniffing position task. More than 90 degree HNM is 0 points, 90 degrees HNM is 1 point, and less than 90 degree HNM is 2 points. BMI of a patient can also complicate the procedure. Therefore, regular BMI is given 0 points, BMI rating of obese patient is 1 point, and BMI rating equal to severely obese is 2 points.

'No incisor' or 'any prominent incisors' is the base scenario, which is 0 points for the prominent incisors assessment factor. The patient with upper incisors protruded 0 to 0.5 cm more than lower teeth is 1 point, and upper incisors protruded more than 0.5 cm more than lower teeth is 2 points. Inter-incisor gap is the distance between the upper and lower teeth when the mouth is opened wide. Inter-incisor gap more than 5 cm is 0 points, 4 or 5 cm is 1 point, and less than 4 cm is 2 points. ULBT is the ability of lower teeth to bite the upper lip. Lower incisors hiding the mucosa of upper lip is 0 points, the lower incisors partially hiding the mucosa of upper lip is 1 point, and lower incisors are unable to touch the mucosa of upper lip is 2 points. Table 8 shows the factor-based system and Table 9 shows the point-based scoring system.

Although Figure 16 shows the base case for ETI, The moderate case created with our GAML platform is shown in Figure 17. Mallampati class is generated by expanding the uvula in the Y axis, increasing the size of tongue with respect to the anatomy models adjacent to it. Inter-incisor gap was generated by moving the upper and lower teeth closer together.

## 6 | CONCLUSION

The use of 3D anatomical models became very popular among the medical educators in the last decade due to the efficiency provided in depicting anatomical structures in comparison to cadaver use. Enhanced understanding of the complex structures in human anatomy necessitates the need for sophisticated software tools and environments for defining spatially complex relationships between structures by modeling challenging procedures in a virtual environment. Therefore, within the scope of this study, we have created GAML as an online platform to ease development in creating variations of 3D models that require numerous modifications. GAML provides constraint mechanism that enable incorporation of the anatomical constraints in 3D modeling. Our framework ensures validity of the generated models by fulfilling the constraints as part of the nonlinear optimization model. In this study, we used GAML to create difficult scenarios for training the airway management techniques for our virtual simulator.

## 7 | FUTURE WORK

Our future goal is to increase the variety of modifiers that can be applied to different anatomy. We further plan to increase the variety of constraints and support custom constraint creation to enhance modeling that will better reflect complexity of human anatomy in our optimization model. We will also improve the user friendliness of the GUI by introducing more interactive elements (e.g. including sliders and other simple tools) to make the adjustments for the constraints or operators. With the help of increased numbers of modifiers, custom constraints, and user friendly enhancements, we want to create a versatile system that can be used to create variations in any anatomy.

## ACKNOWLEDGEMENT

The Authors would like to thank Drs Suvranu De, Kathryn L Butler, Marc A deMoya and Stephanie B Jones for their valuable contributions in this study. This publication was made possible by the Arkansas INBRE program, supported by a grant from the National Institute of General Medical Sciences, (NIGMS), P20 GM103429 from the National Institutes of Health. This project was also supported by National Institutes of Health (NIH) Grant NIH/NHLBI 1R01HL119248-01A1, NIH/NIBIB 2R01EB005807, 5R01EB010037, 1R01EB009362, and 1R01EB014305.

## REFERENCES

1. Cotin S, Shaffer DW, Meglan DA, et al. CAML: a general framework for the development of medical simulation systems. In AeroSense 2000 [Internet]. International Society for Optics and Photonics; 2000 [cited 2016 Sep 2]. 294–300. Available from: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=904099
2. Cartwright R, Adzhiev V, Pasko AA, et al. Web-based shape modeling with HyperFun. *IEEE Comput Graph Appl*. 2005;25(2):60-69.
3. Morkel C, Bangay S. Procedural modeling facilities for hierarchical object generation. In Proceedings of the 4th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa [Internet]. ACM; 2006 [cited 2016 Sep 2]. 145–154. Available from: http://dl.acm.org/citation.cfm?id=1108614
4. Cutler B, Dorsey J, McMillan L, *et al.* A procedural approach to authoring solid models. In *ACM Transactions on Graphics (TOG) [Internet]*. ACM; 2002 [cited 2016 Sep 2]. 302–311. Available from: http://dl.acm.org/citation.cfm?id=566581
5. Lohikoski-Håkansson L, Rudén E. Optimization of 3D game models: a qualitative research study in unreal development kit. 2013 [cited 2016 Dec 1]; Available from: http://www.diva-portal.org/smash/record.jsf?pid=diva2:708048
6. Pighin F, Szeliski R, Salesin DH. Resynthesizing facial animation through 3d model-based tracking. In Computer Vision, 1999 The Proceedings of the Seventh IEEE International Conference on [Internet]. IEEE; 1999 [cited 2016 Dec 1]. 143–150. Available from: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=791210
7. Kim DO, Kang HS, Kim JH, Min BG. Virtual anatomy and movement of lower extremities using virtual reality modeling language. *J Digit Imaging*. 2000 May;13(Suppl 1):238-240.
8. Carey R, Bell G. The annotated VRML 2.0 reference manual [Internet]. Addison-Wesley Longman Ltd; 1997 [cited 2016 Sep 12]. Available from: http://dl.acm.org/citation.cfm?id=270060

9. Kaus MR, von Berg J, Weese J, et al. Automated segmentation of the left ventricle in cardiac MRI. *Med Image Anal*. 2004 Sep;8(3):245-254.
10. Sierra R, Bajka M, Székely G. Evaluation of different pathology generation strategies for surgical training simulators. In: *International congress series [Internet]*. Elsevier; 2003 [cited 2016 Oct 5]. 376–381. Available from: http://www.sciencedirect.com/science/article/pii/S0531513103004473
11. Havemann S, Fellner DW. Generative Mesh Modeling. [Internet]. University of Braunschweig-Institute of Technology; 2005 [cited 2016 Jul 27]. Available from: http://generalized-documents.org/CGVold/DigitalLibrary/publications/TechnicalReports/bs/TR-tubs-cg-2003-01.pdf
12. Demirel D, Butler KL, Halic T, et al. A hierarchical task analysis of cricothyroidotomy procedure for a virtual airway skills trainer (VAST) simulator. *Am J Surg*. 2016 Sep;212(3):475–484.
13. Halic T, Ahn W, De S. A framework for web browser-based medical simulation using WebGL. In MMVR [Internet]. 2012 [cited 2015 Oct 6]. 149–155. Available from: https://books.google.com/books?hl=en&lr=&id=JYk1nUgIMlEC&oi=fnd&pg=PA149&dq=sofmis+halic&ots=Muubk2myCf&sig=EOpTiPDyAO2jr0V6c2cOhDknpkA
14. Demirel D, Yu A, Halic T, Kockara S. Web based camera navigation for virtual pancreatic cancer surgery: whipple surgery simulator (VPanSS). In 2014 IEEE Innovations in Technology Conference (InnoTek). 2014. 1–8.
15. Halic T, Venkata SA, Sankaranarayanan G, et al. A software framework for multimodal interactive simulations (SoFMIS). *Stud Health Technol Inform*. 2011;163:213-217.
16. Donnelly C, Stallman R. *Bison, Free Softw Found*. 1995;51:2110-1301.
17. Lalonde WR, Lee ES, Horning JJ. An LALR(k) parser generator. In: *Proc. IFIPS Congress, 1971*. Ljubljana, Yugoslavia: North-Holland Publishing Co., Amsterdam; 1972:513-518.
18. Powell MJ. A fast algorithm for nonlinearly constrained optimization calculations. In *Numerical analysis [Internet]*. Springer-Verlag Berlin Heidelberg: Springer; 1978 [cited 2016 Sep 14]. 144–157. Available from: http://link.springer.com/content/pdf/10.1007/BFb0067703.pdf
19. Powell MJ. A direct search optimization method that models the objective and constraint functions by linear interpolation. In *Advances in Optimization and Numerical Analysis* [Internet]. Springer Netherlands: Springer; 1994 [cited 2016 Nov 28]. 51–67. Available from: http://link.springer.com/chapter/10.1007/978-94-015-8330-5_4
20. Demirel D, Yu A, Halic T, Sankaranarayanan G, et al. Virtual airway skills trainer (VAST) simulator. *Med Meets Virtual Real 22 NextMedMMVR22*. 2016;220:91
21. Benumof JL. The ASA difficult airway algorithm: new thoughts and considerations. In: *Handb Difficult Airw Manag*. Phila Pa: Churchill Livingstone; 2000:31-48.
22. Apfelbaum JL, Hagberg CA, Caplan RA, et al. Practice Guidelines for Management of the Difficult Airway: An Updated Report by the American Society of Anesthesiologists Task Force on Management of the Difficult Airway. *Anesthesiology*. 2013 Feb;118(2):251-270.
23. Seo S-H, Lee J-G, Yu S-B, et al. Predictors of difficult intubation defined by the intubation difficulty scale (IDS): predictive value of 7 airway assessment factors. *Korean J Anesthesiol*. 2012 Dec;63(6):491-497.
24. Möller T, Trumbore B. Fast, minimum storage ray-triangle intersection. *J Graph Tools*. 1997 Oct;2(1):21-28.


**How to cite this article:** Demirel D, Yu A, Baer-Cooper S, Halic T, Bayrak C. Generative Anatomy Modeling Language (GAML). *Int J Med Robotics Comput Assist Surg*. 2017;e1813. https://doi.org/10.1002/rcs.1813

## APPENDIX A

```
\s+ /* skip whitespace */.
    ([0–9] + (“.”[0–9]+)?\b)|(“.”[0–9]+) return “NUMBER”
    “a little”|”A LITTLE” return “LITTLE”
    “a lot”|”A LOT” return “LOT”
    “moderate”|”MODERATE”return “MODERATE”
    “a bit”|”A BIT” return “ABIT”
    “further”|”FURTHER” return “FURTHER”
    “less”|”LESS”return “LESS”
    “more”|”MORE”return “MORE”
    “much more”|”MUCH MORE”return “MORE”
    “far”|”FAR”return “FAR”
    “near”|”NEAR”return “NEAR”
    “duplciate”|”DUPLCIATE”return “DUPLCIATE”
    “tesselation”|”TESSELATION”return “TESSELATION”
    “smooth”|”SMOOTH”return “SMOOTH”
    “displaceout”|”DISPLACEOUT”return “DISPLACEOUT”
    “displacein”|”DISPLACEIN”return “DISPLACEIN”
    “&” return “AND”
    “!CREATETEARATPOINT”|”CREATETEARATPOINT”|”
createtearatpoint”return “CCREATETEARATPOINT”
    “!CREATETEAR”|”CREATETEAR”|”createtear”return
“CREATETEAR”
    “!PLACEJOINTATPOINT”|”PLACEJOINTATPOINT”|”
placejointatpoint”return “PLACEJOINTATPOINT”
    “!PLACEJOINT”|”PLACEJOINT”|”placejoint”return “PLACEJOINT”
    “!PLACEABSJOINT”|”PLACEABSJOINT”|”placeabsjoint”return
“PLACEABSJOINT”
    “!PLACEFLEXJOINT”|”PLACEFLEXJOINT”|”placeflexjoint”return
“PLACEFLEXJOINT”
    “!PLACEANGLEJOINT”|”PLACEANGLEJOINT”|”
placeanglejoint”return “PLACEANGLEJOINT”
    “!FLEXIBILITY″|”FLEXIBILITY″|”flexibility”return “FLEXIBILITY”
    “!LINKWITH”|”LINKWITH”|”linkwith”return “LINKWITH”
    “!IDENTIFY AS”|”IDENTIFY AS”|”identify as”return “IDAS”
    “!IDENTIFIED AT POINT AS”|”IDENTIFIED AT POINT
AS”|”identified at point as”.
    return “IDASPOINT”
    “!SELECTION SIZE”|”SELECTION SIZE”|”selection size” return
“SELECTIONSIZE”
    “Irradiate”|”IRRADIATE”|”irradiate” return “IRRADIATION”
    “SCARSIZE”|”scarsize”|”scar size”|”SCAR SIZE”return “SCARSIZE”
    “IRRADIATIONSIZE”|”irradiationsize”|”irradiation
size”|”IRRADIATION SIZE”.
    return “IRRADIATIONSIZE”
    “!SCAR”|”SCAR”|”scar” return “SCAR”
    “!ENLARGE UNTIL”|”ENLARGE UNTIL”|”enlarge until” return
“ENLARGEUNTIL”
    “!ENLARGE”|”ENLARGE”|”enlarge” return “ENLARGE”
    “!ADJUSTATPOINT”|”ADJUSTATPOINT”|”adjustatpoint” return
“ADJUSTATPOINT”
```

```
"!ADJUST"|"ADJUST"|"adjust" return "ADJUST"
"!ADJUSTOVERALL"|"ADJUSTOVERALL"|"adjustoverall" return "ADJUSTOVERALL"
"!AND"|"AND"|"and" return "AND"
"," return ","
"!ADD"|"ADD"|"add" return "ADD"
"!REMOVE"|"REMOVE"|"remove" return "REMOVE"
"!LINK"|"LINK"|"link" return "LINK"
"!SELECT"|"SELECT"|"select" return "SELECT"
"!DESELECT"|"DESELECT"|"deselect" return "DESELECT"
"!MUSCLE"|"MUSCLE"|"muscle" return "MUSCLE"
"!JOINT"|"JOINT"|"joint" return "JOINT"
"!VEIN"|"VEIN"|"vein" return "VEIN"
"!ARTERY"|"ARTERY"|"artery" return "ARTERY"
"!BONE"|"BONE"|"bone" return "BONE"
"!LIGAMENT"|"LIGAMENT"|"ligament" return "LIGAMENT"
"!FAT"|"FAT"|"fat" return "FAT"
"!TEETH"|"TEETH"|"teeth" return "TEETH"
"!SKIN"|"SKIN"|"skin" return "SKIN"
"!CARTILAGE"|"CARTILAGE"|"cartilage"return "CARTILAGE"
"MOVE CORONAL"|"move coronal"|"move x"|"MOVE X"|"!MOVE X" return "MOVECORONAL"
"MOVE SAGITTAL"|"move sagittal"|"move y"|"MOVE Y"|"!MOVE Y"
return "MOVESAGITTAL"
"MOVE TRANSVERSE"|"move transverse"|"move z"|"MOVE Z"|"!MOVE Z".
return "MOVETRANSVERSE"
"SCALE CORONAL"|"scale coronal"|"scale x"|"SCALE X"|"!SCALE X"
return "SCALECORONAL"
"SCALE SAGITTAL"|"scale sagittal"|"scale y"|"SCALE Y"|"!SCALE Y"
return "SCALESAGITTAL"
"SCALE TRANSVERSE"|"scale transverse"|"scale z"|"SCALE Z"|"!SCALE Z"
return "SCALETRANSVERSE"
"ROTATE CORONAL"|"rotate coronal"|"rotate x"|"ROTATE X"|"!ROTATE X"
return "ROTATECORONAL"
"ROTATE SAGITTAL"|"rotate sagittal"|"rotate y"|"ROTATE Y"|"!ROTATE Y"
return "ROTATESAGITTAL"
"ROTATE TRANSVERSE"|"rotate transverse"|"rotate z"|"ROTATE Z"|"!ROTATE Z"
return "ROTATETRANSVERSE"
"MOVE"|"move" return "MOVE"
"SCALE"|"scale"|"!SCALE" return "SCALE"
("#"[0–9]+)\b
return "HEIR"
[A-z] + \b.
return "MODEL"
"= > ".
return "MARK"
"*"
return "*"
"/"
return "/".
"-"
return "-"
"+"
return "+".
"^"
return "^".
"%"
return "%".
"("
return "(".
")"
return ")".
"E"
return "E".
<<EOF>>.
return "EOF".
.
return "INVALID".
/lex.
/* operator associations and precedence */.
%left "+" "-".
%left ","
%left "*" "/".
%left "^".
%left UMINUS.
%left LOT.
%left MODERATE.
%left LITTLE.
%left ABIT.
%left FURTHER.
%left AND.
%left LESS.
%left PLACEJOINT.
%left MORE.
%left PLACEJOINTATPOINT.
%left MORE.
%left LINK.
%left FAR.
%left DUPLCIATE.
%left TESSELATION.
%left FLEXIBILITY.
%left SMOOTH.
%left NEAR.
%left DISPLACEOUT.
%left ENLARGE.
%left DISPLACEIN.
%left ENLARGEUNTIL.
%left IRRADIATION.
%left CREATESCAR.
%left CREATESCARATPOINT.
%left LINKWITH.
%left SCAR.
%left ATPOINT.
%left IDASPOINT.
%left IRRADIATIONSIZE.
```

```
%left SCARSIZE.
%left SELECTIONSIZE.
%right "!"
%right "%".
%left ADJUSTOVERALL.
%left ADJUSTATPOINT.
%left ADJUST.
%left SKIN.
%left REMOVE.
%left ADD.
%left SELECT.
%left DESELECT.
%left MOVECORONAL.
%left MOVESAGITTAL.
%left MOVETRANSVERSE.
%left SCALE.
%left SCALECORONAL.
%left SCALESAGITTAL.
%left SCALETRANSVERSE.
%left ROTATECORONAL.
%left ROTATESAGITTAL.
%left ROTATETRANSVERSE.
%left MARK.
%left MUSCLE.
%left JOINT.
%left VEIN.
%left ARTERY.
%left BONE.
%left LIGAMENT.
%left FAT.
%left TEETH.
%left SKIN.
%left CARTILAGE.
%left IDAS.
```