ELSEVIER

Artificial Intelligence in Medicine 25 (2002) 227–245

Artificial Intelligence in Medicine

www.elsevier.com/locate/artmed

Combining biometric and symbolic models for customised, automated prosthesis design

S. Modgil[a,*], T.J. Hutton[a], P. Hammond[a], J.C. Davenport[b]

[a]*Department of Biomedical Informatics, Eastman Dental Institute for Oral Health Care Sciences, University College London, 256 Gray's Inn Road, London WC1X 8LD, UK*

[b]*School of Dentistry, The University of Birmingham, St. Chad's Queensway, Birmingham B4 5NN, UK*

Received 18 May 2001; received in revised form 4 February 2002; accepted 20 February 2002

Abstract

In a previous paper [Artif. Intell. Med. 5 (1993) 431] we described RaPiD, a knowledge-based system for designing dental prostheses. The present paper discusses how RaPiD has been extended using techniques from computer vision and logic grammars. The first employs point distribution and active shape models (ASMs) to determine dentition from images of casts of patient's jaws. This enables a design to be customised to, and visualised against, an image of a patient's dentition. The second is based on the notion of a path grammar, a form of logic grammar, to generate a path linking an ordered sequence of subcomponents. The shape of an important and complex prosthesis component can be automatically seeded in this fashion. Combining these models now substantially automates the design process, beginning with a photograph of a dental cast and ending with an annotated and validated design diagram ready to guide manufacture.

Keywords: Prosthesis; Design; Automated; Active shape models; Logic grammars

1. Introduction

Removable partial dentures (RPDs) replace missing teeth and related tissues in individuals who have some natural teeth. A typical RPD annotated with a description of its main components is shown in Fig. 1. The complete design and manufacturing process is summarised in Fig. 2. It begins with intra and extra oral examinations and the production of dental casts of a patient's jaws (Fig. 2 stage 1). The casts are surveyed to identify important three dimensional (3D) information that influences RPD design; especially the

* Corresponding author. Tel.: +44-20-7915-2343; fax: +44-20-7915-2303.
E-mail addresses: s.modgil@eastman.ucl.ac.uk (S. Modgil), t.hutton@eastman.ucl.ac.uk (T.J. Hutton), p.hammond@eastman.ucl.ac.uk (P. Hammond), john.davenport@btclick.com (J.C. Davenport).

0933-3657/02/$ – see front matter
PII: S0933-3657(02)00026-X

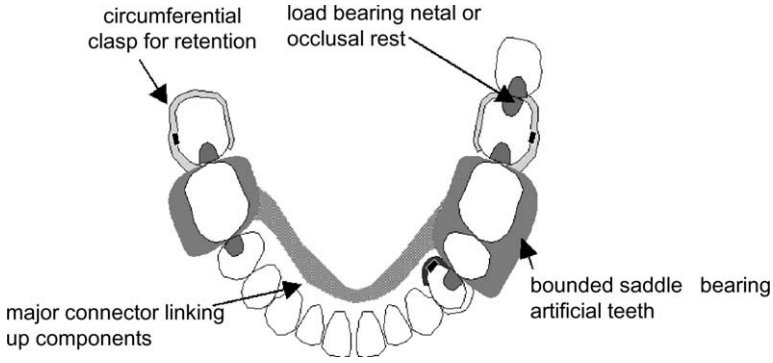
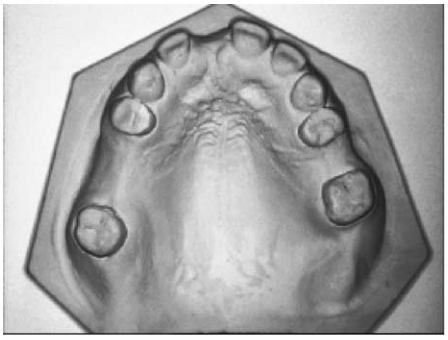
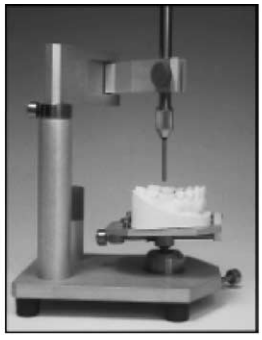


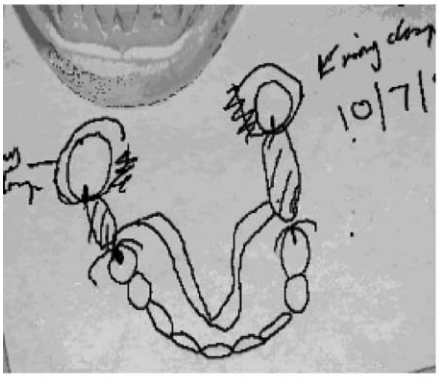
Fig. 1. Annotated RPD with description of components.



Stage 1: Typical dental cast of the upper jaw



Stage 2: Surveying of cast to generate 3D information to guide placement of components



Stage 3: Design prescription (of poor quality)

Stage 4: Lost wax manufacturing

Fig. 2. Stages in the design and manufacture of an RPD.

location of elements to maintain its position during speech and when eating (stage 2). Then, the dentist produces a paper-based design to guide manufacture (stage 3). Finally, the paper design and the plaster casts are sent to a laboratory where a dental technician overlays the latter with a wax mark up and mould (stage 4). For a metal RPD, the introduction of molten cobalt chromium causes the wax to melt in the mould and form a framework. Artificial teeth and gum-like flanges are then added where missing teeth are to be replaced as specified in the design.

In the UK alone, 650,000 RPDs are manufactured annually, with a cost to the UK National Health Service of over £50 million per annum. Even with advances in sophisticated dental bridges and osseo-integrated implants, there will be a strong demand for RPDs for decades to come. Extended lifespan, increased wealth and retention of more, but not necessarily all, natural teeth also contribute to continued RPD usage [17]. RPD design requires a detailed knowledge of design principles and clinical factors. Evidence of poor RPD design (an example of which is shown in Fig. 2 stage 3) by general dental practitioners in the USA, Australia, Scandinavia and UK [7], and the unethical delegation of design to dental technicians, stimulated the development of RaPiD, a prototype knowledge-based system for designing RPDs. RaPiD uses techniques from constraint-based critiquing, logic databases and declarative graphics [10].

RaPiD assists in stage 3 by providing expert guidance during manual design on the computer. It is implemented in DECADE, a generic environment for building intelligent design applications [11]. DECADE provides a computer-aided design (CAD) style interface in which direct manipulation of graphical depictions of design components is interpreted as an update to a logic database of component relationships. These updates are checked against a set of integrity constraints representing the expert design knowledge [6]. Unacceptable design updates are critiqued and more acceptable alterations are suggested. Fig. 1 shows an example of an RPD designed using RaPiD. Apart from designing RPDs, RaPiD can also be used to: visualise treatment options for patients; print work authorisations prior to manufacture; communicate designs electronically; offer education and training in RPD design, and provide records for clinical audit. Recently, RaPiD has been commercialised under combined government and industrial funding (http://www.tcsonline.org.uk/). This close integration of RaPiD with an existing patient information system enhances its usability in the clinical setting.

Under funding from the UK National Health Service, RaPiD is being evaluated by 36 dentists in their practices and by six commercial dental laboratories who manufacture RPDs. Although the trial is incomplete, three notable requirements (previously suggested anecdotally) have received direct support, suggesting the following changes.

Firstly, RaPiD's design rules were initially compiled from a single expert [6]. This over-reliance on one individual's view was recognised and subsequently addressed by a survey of 70 or so expert prosthetic dentists who were asked to indicate support for potential design rules obtained from an extensive literature review. Only those receiving majority support were included in the revised knowledge base [4,5].

Secondly, RaPiD employed a generic representation of dentition and so was unable to produce a design that is customised to an individual patient (see Fig. 1). In Section 2.1, we illustrate the use of point distribution and active shape models (ASMs) [3] for modelling the variation in shape of the dentition across the population. ASMs can be used to locate the

outlines of teeth automatically from a photograph of the dental cast. This information can then be used to customise the design of the RPD.

Thirdly, RaPiD users have often requested greater support for automating the design process while maintaining the ability to amend a design and invoke its validation against the embedded design expertise. In the RaPiD prototype, the user manually draws the major connector connecting the RPD components (Fig. 1). However, this is an awkward task, and users often design connectors that do not conform to complex design rules governing both order and form of linkage. In Section 2.2 we introduce path grammars. These are extensions of logic grammars, and are used to generate the contour of a path connecting subcomponents automatically according to design rules. This path then seeds the shape of the major connector.

Section 3 summarises the results of combining these techniques and the final section speculates on future research involving design in 3D and the potential for computer-aided manufacture. The application of the new biometric and symbolic models described in this paper, enables stage 3 in Fig. 2 to be replaced almost entirely by the capture of a digital image of the dental cast and the automatic generation of a basic design.

2. Materials and methods

2.1. Active shape models

In ASMs [3] shape analysis methods are used to derive a statistical model of permissible shape configurations from a collection of annotated images. This model is then used to help search new images for matching structures. ASMs have previously been used successfully to tackle other problems in the field of dentistry [12,13].

The model of variation is derived from a training set of examples, in our case a set of occlusal photographs of casts of teeth. Fifty-four upper and 61 lower casts were selected to demonstrate the variation in arch width and tooth presence as well as tooth position, rotation and shape. The images were acquired using a Sony DC50 digital camera with a Coreco grabber board; a Dolan–Jenner ringlight was used to give reproducible lighting. Such hardware is relatively inexpensive and easy to use.

The dentition of a patient is represented, as in RaPiD, by a set of polygonal tooth outlines comprising in total 444 vertices and referred to as a template. Each image in the training set was hand-annotated by manually moving each template point into place (see Fig. 3 for two examples).

To model the variation statistically requires data on all the points in the template. Where teeth are missing in the training set images, standard data patching methods can be used to estimate locations that are most consistent with the model; missing teeth in Fig. 3 were patched-in in this way.

Following [3], the model of shape variation is computed in a two-stage process. Firstly, the set of templates are aligned with each other in a least-squares sense using the generalised Procrustes algorithm [8,9]:

1. Align (rotate, scale and translate) all the templates with the first template.
2. Compute the mean of the set of templates.
3. Align the mean template with the first template.

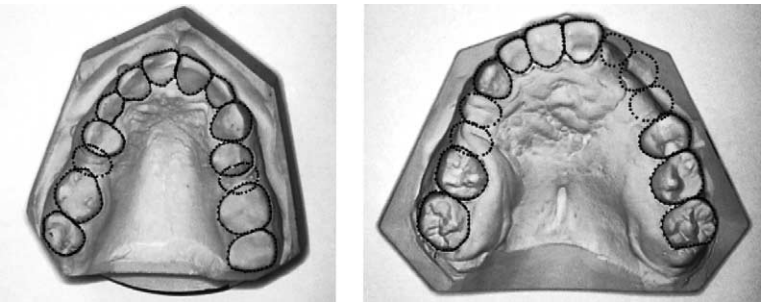


Fig. 3. Two example dental cast images with the shape templates overlaid.

4. Align all the templates to this mean.
5. Repeat from step 2 until no further change is observed.

The co-ordinates of the 148 points in a template form a vector of length 296:

$$\boldsymbol{v} = [x_1 y_1 \ \ldots \ x_{148} y_{148}]^T \tag{1}$$

After alignment, the second stage is to analyse these residuals to build a statistical model of the deformation. The covariance matrix, $\boldsymbol{S}$, is computed by using:

$$\boldsymbol{S} = \sum_{i=1}^{n} \frac{(\boldsymbol{v}_i - \boldsymbol{v}_{\text{mean}})(\boldsymbol{v}_i - \boldsymbol{v}_{\text{mean}})^T}{(n-1)} \tag{2}$$

where n is the number of examples in the training set. The decomposition of $\boldsymbol{S}$ gives eigenvectors $\phi_1 \ldots \phi_n$ and eigenvalues $\lambda_1 \ldots \lambda_n$. The eigenvectors are sorted by the magnitude of their corresponding eigenvalues. The size of an eigenvalue as a proportion p_j of the sum of all the eigenvalues determines the proportion of variation associated with each eigenvector:

$$p_j = \frac{\lambda_j}{\sum_{i=1}^{n} \lambda_i} \tag{3}$$

Linear sums of the eigenvectors can be added to the shape vector for the mean template to create new shapes:

$$\boldsymbol{v}_{\text{new}} = \boldsymbol{v}_{\text{mean}} + \boldsymbol{\Phi b} \tag{4}$$

where $\boldsymbol{\Phi}$ is the matrix of eigenvectors and $\boldsymbol{b}$ is a set of weights.

The active shape model thus consists of a mean template plus a set of deformations or modes of shape variation. A crucial feature of the model is that as long as reasonable limits are imposed, any combination of deformations yields a dentition that is plausible, i.e. likely to have come from the population from which the training set was sampled. These limits are derived from the probability distribution of the training set, typically each $\boldsymbol{b}_i$ is varied within three S.D.

As long as the training set covers enough variation, any reasonable dentition can be synthesised using a deformable template. Thus, the model can be used to fit to unseen examples.

2.1.1. Determining tooth outlines in unseen images

To extract the tooth outlines in a new image, a process of interpretation by synthesis is used. The deformable template is overlaid on the image and iteratively adjusted until it best matches the structures seen within the image. To achieve this, each template point has a model of the local grey-level appearance in the training images; for example, a point on the edge of a tooth might expect a lighter colour on one side and a sudden change to a darker colour on the other. This grey-level model is derived automatically from the training set by sampling each of the images around each template point. For more details see for example [3].

Together these points make a new template shape but probably one that is misshapen because it is not constrained—the inevitable matching errors will cause the template points to become disjoint. This is overcome by only permitting certain kinds of deformation, those that are consistent with the training set. The parameters $\boldsymbol{b}$ for the closest valid shape in the model to a misshapen template $\boldsymbol{v}$ can be found using:

$$\boldsymbol{b} = \boldsymbol{\Phi}^T(\boldsymbol{v} - \boldsymbol{v}_{\text{mean}}) \tag{5}$$

and limiting each b_i to be in the three S.D. range:

$$-3\sqrt{\lambda_i} < \boldsymbol{b}_i < 3\sqrt{\lambda_i} \tag{6}$$

before applying (4) to obtain the replacement shape.

The initial placement of the template is important—if the template starts off too far from the target the algorithm may not produce an acceptable fit. We ensure that all the images are acquired in a similar way, with the cast in the centre of the photograph and always oriented the same way. The mean location, rotation and scaling of the manually fitted templates in the training set images can then be used as the starting point for the search. This is done because the ASM technique is a local optimizer—it will only converge correctly if initiated reasonably close to the target.

The fitting process is iterated until no further movement is observed. The final configuration of the template on the image yields the outlines of the teeth. These outlines form the input to the next stage, where they are used to create a customised RPD design.

2.2. Semi-automated design of dental prostheses

Fig. 4 shows just a few of the wide range of the 100,000 or so possible RPD designs. In particular, the designs illustrate the shape variation of the most important component, the major connector (shown shaded). The prime function of the major connector is to link together all other components of the prosthesis: the acrylic saddles, the metal rests, and the retention clasps.

The RaPiD prototype supports the automatic generation and positioning of some components (rests and saddles) for straightforward clinical cases, by simply invoking default positions, shapes and sizes provided by the expert prosthetists. These can be modified manually when required. Retention clasps require 3D information to determine both their precise location on a tooth, and their length. Hence, they are

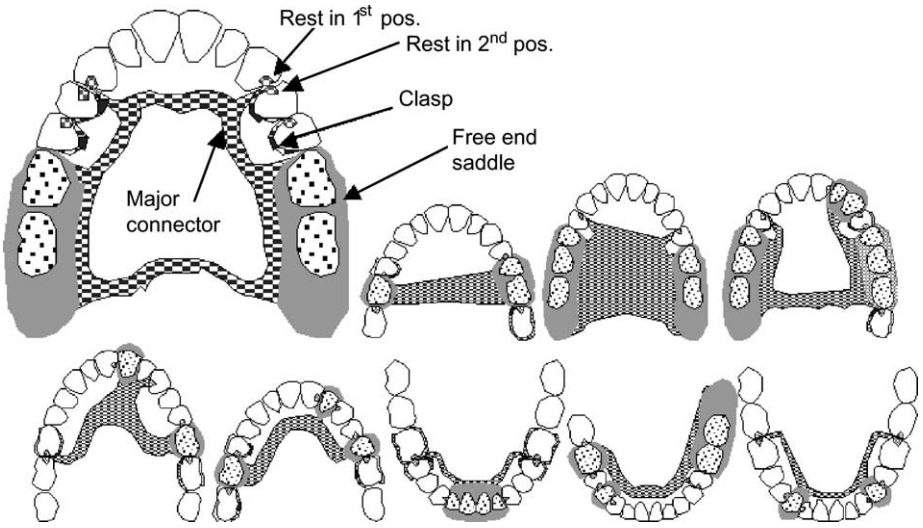


Fig. 4. Collection of RPD designs showing wide variation in shape of the major connector.

currently placed manually. The RaPiD prototype does not support automatic generation of the major connector component. The user is expected to describe the connector's outline by dragging the mouse, successively clicking at points where the outline changes course. Users of RaPiD have found this method to be fiddly, time-consuming and imprecise, and generally expressed a preference for more automation followed by user refinement. Furthermore, the design rules (constraints) governing connector shape were not animated as critiques, so that designed connectors often did not conform to these rules. With this in mind, we next describe our approach to automating the generation of the major connector.

2.2.1. Logic grammars for generation of major connector outlines

Logic grammars [16] are traditionally used for compact descriptions of the constituent structure (syntax) of languages through sets of grammar rules. Sentences are described as linear sequences of words which can be decomposed into semantically meaningful subsequences, enabling both parsing and generation of sentences. Analogously, an arch of teeth, when read clockwise, can be interpreted as a linear sequence of different types of object, decomposable into semantically meaningful subsequences.

Parsing a sequence of teeth can thus identify subsequences, each of which exert specific local constraining effects on the ''path'' taken by the major connector as it passes from one RPD component to the next. Observation of the expert's manual drawing of the major connector outline inspired this parsing approach. Typically, components are connected in a clockwise fashion with care being taken on how the outline enters and exits individual and sequences of components. In this section, we briefly describe the use of grammars to define the major connector shape. In the next section we show how the more generic notion of a path grammar improves on and subsumes this approach.

Table 1
Icons and tokens in the connector grammar

Tooth type	Icon	Token	Tooth type	Icon	Token
Natural		n	Rest in second pos		n2
Artificial		a	Two rests		n12
Missing		m	Periodontally compromised	X	x
Rest in first pos		n1	Onlay		o

Magnified arch in Fig. 4 represented as sequence of icons

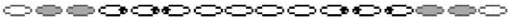


Table 1 summarises the eight icons and symbolic tokens used to represent the elements in the partially dentulous arch. The upper arch magnified and annotated in Fig. 4, when read in a clockwise direction, corresponds to the sequence of icons shown in Table 1, and to the list of symbolic tokens: [m, a, a, $n2$, $n2$, $n1$, n, n, n, n, $n2$, $n1$, $n1$, a, a, m]. The token sequence can be parsed by a definite clause grammar (dcg) expressed in the logic programming language Prolog. Some rules of the basic dcg are shown below.

arch → free_end_start, nat_art_seq
arch → free_end_start, nat_art_seq, free_end_finish
arch → missing, nat_art_seq, free_end_finish
arch → missing, nat_art_seq, missing

missing → [ ]
missing → [m], missing

free_end_start → free_end_initiator, saddle
free_end_finish → saddle, free_end_initiator

free_end_initiator → [a]
free_end_initiator → [m]

nat_art_seq → natseq
nat_art_seq → natseq, saddle, nat_art_seq

natseq → nat_subseq
natseq → nat_subseq, natseq

nat_subseq → minor_connector_pair
nat_subseq → natural

saddle → [a]
saddle → [a], saddle

minor_connector_pair → [$n2$, $n1$]

natural → [n]
natural → [$n1$]
natural → [$n2$]
natural → [$n12$]
natural → [o]
natural → [x]

These rules can be used to parse a complete arch represented as symbolic tokens, progressively parsing subsequences in a clockwise direction. Logic grammars are typically extended so that in addition to parsing a sentence they produce a parse tree capturing the overall structure of the sentence in a single term. We can extend the above grammar so as to generate a parse tree in the form of a list of expressions describing boundary or "seed" points related to features (typically one or two) of the subsequences found. For example,

the grammar rules listed above, parsing subsequences of natural and artificial teeth, might be annotated as follows:

nat_art_seq(S1, S2) → natseq(S1, S2)
nat_art_seq(S1, S3) → natseq(S1, [inner(entry(first(A))), entry(first(A)), exit(last(A)), inner(exit(last(A)))|S2]), saddle(A), nat_art_seq(S2, S3)
saddle([*a*]) → [*a*]
saddle([*a*|A]) → [*a*], saddle(A)

Fig. 5a depicts part of an arch comprising a natural tooth, a pair of artificial teeth and another natural tooth. In order to connect up the saddle bearing artificial teeth, the outline of the major connector needs to pass through the points (U–Z). The symbolic specification of points V–Y Fig. 5b) can be generated by the extended grammar rules above. The functions used in the parse tree have obvious interpretations: first and last compute the first and last elements in a subsequence; entry and exit denote the points through which a connector outline enters and exits a tooth or component subsequence; and inner computes an inner projection along a ''normal'' to the outline.

The rest of the connector outline grammar is extended in the same manner with expressions representing other seed points. If these expressions are computed they generate the co-ordinates of seed points forming a crude outline for part of the connector boundary. However, prior to computation, these expressions can be manipulated symbolically to give more refined shapes or be altered to comply with design rules governing the shape of the connector. For example, our survey of design experts confirmed support for the following connector rule: A maxillary connector should cover the gingival margin when a single tooth separates a minor connector from a distal extension base (free end saddle).

This rule is more easily appreciated in diagrammatic form in Fig. 5c and d. Such rules can be applied after parsing takes place, so that they are applied to the higher level structures of an extended parse tree. The adjacent components, free end saddle and minor connector pair (pair of natural teeth with adjacently placed rests) are recognised in the parse tree, and the sequence of seed points A–G is modified to the sequence A, F, G. Only after such design rules are applied need the seed point expressions be computed.

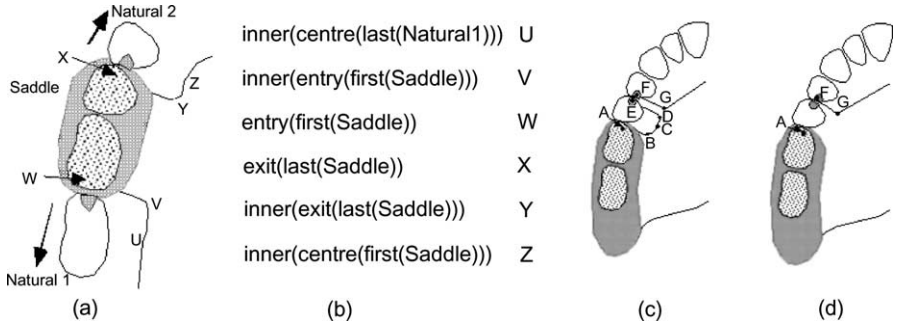


Fig. 5. Graphical (a) and symbolic form (b) of seed points; visualisation of a connector rule (d is preferred to c).

Notice that, encoded in the grammar are the structural features looked for in the extended parse tree to validate application of a design rule. For example, compliance with the connector rule above requires explicitly encoding in the grammar that a subsequence of natural teeth may be defined by a minor connector pair (nat_subseq → minor_connector_pair). This ''object level'' encoding of features to enable application of design rules entails a number of disadvantages.

Firstly, in practice a significant number of design rules may apply, so that the above approach implies considerable complexity in the specification of the grammar, with concomitant lack of transparency and difficulty in maintenance.

Secondly, when validating the knowledge base, not all proposed rules received majority support (design styles vary between individuals depending on training and professional experience). Thus, one would want potential users to be able to configure the design rules, ''turning rules on and off'' according to their preferences. Given this requirement, then accounting for all the design rules in the object level specification of the grammar is a significant source of inefficiency.

Thirdly, another source of inefficiency arises because the global character of design rules is not reflected by the local characterisation of structural features that in combination validate their application. For example, given a sequence of tokens, it is only necessary to make use of the grammar rule nat_subseq → minor_connector_pair in the parse of some subsequence of natural teeth, if the previous tokens parsed define a saddle (a sequence of artificial teeth). Only in this case does the connector rule apply, and so only in this case is it necessary to record this feature in the extended parse tree generated. However, as defined, any subsequence of natural teeth in a given token sequence is parsed using this grammar rule.

These observations motivated the development of path grammars, which are a generalisation and extension of the above approach. Their use for generating major connector outlines overcomes the above mentioned disadvantages. Furthermore, path grammars provide a more flexible and generic approach potentially applicable in other domains.

2.2.2. Path grammars

Path grammars provide a means for automated path generation, connecting sequences of objects, subject to design rules/constraints. Each object in the domain is assigned a unique object type and a set of domain specific seed points imposing local constraints on how the path traverses an object of that type. The sets of seed points of each distinct pair of neighbouring objects are composed according to an operation pre-determined by their object types. Thus, given a sequence of objects, successive composition of each adjacent pair's seed points determines a connecting path.

Generic schemes are also provided for specifying the application of global design rules, from hereon referred to as ''path modification (pm) rules''. These are automatically translated to grammar rules, so that a sequence of objects can be successively parsed by each pm rule. A successful parse indicates that the pm rule applies to the sequence. Upon a successful parse, the composition operations on seed point sets of neighbouring objects are appropriately modified resulting in a variation on the default path. In what follows, we describe path grammars in some detail. We require the following definitions.

Definition 1 For a domain D, let:

1. T denote a set of types.
2. D_τ, $\tau \in T$ denote the set of objects of type τ.
3. $O_C = \cup_{\tau \in C} D_\tau$, the union of sets of objects for each $\tau \in C$, where $C \subseteq T$.
4. $S_C = \{\text{start}\} \otimes \otimes_{i=1}^{n} O_C \otimes \{\text{end}\}$, the sequences of length $n+2$ of objects of types in C, $C \subseteq T$, and start and end tokens ($\otimes$ denotes the Cartesian product).
5. Se denote a set of symbolic representations of seed points, and m_τ a τ parameterised mapping m_τ: $D_\tau \rightarrow$ Se. Intuitively, m_τ maps an object X to a default set of points to be navigated by a path passing via X.
6. $\mathscr{C}_\Phi = \{\phi_1, \phi_2, \ldots, \phi_n\}$ denote a set of composition operations (transitions). Each transition ϕ is a tuple of ordered sets $(\{f_1, \ldots, f_m\}, \{g_1, \ldots, g_n\})$, where f_i and g_i are composition functions.

Let S_i, $S_j \in P(Se)$, the powerset of Se. Then, a transition ϕ: $P(Se) \otimes P(Se) \rightarrow P(Se) \otimes P(Se)$ is a mapping such that $\phi((S_i, S_j)) = (f_m(\ldots f_1(S_i)\ldots), g_n(\ldots g_1(S_j)\ldots))$.

We illustrate the above with a simple example in which we imagine a robot planning a path to be taken among a sequence of objects of different types. Associated with each object type is a set of surrounding default seed points which need to be traversed. However, the robot may apply pm rules which if applicable can be used to augment or shorten the path (in terms of the number of points that need to be visited).

Let C be the set of types $\{a, b, c\}$, and τ_i, $i = 1, \ldots, n$, denote an object of type τ, $\tau \in C$. Let Se = {left, right, top, bottom, left_v, right_v, top_v}. Then,

$$m_a(X) = \{\text{left, right, top, bottom}\}$$

$$m_b(X) = \{\text{left\_v, right\_v, top\_v}\}$$

$$m_c(X) = \{\text{left, right, top, bottom}\}$$

Suppose we have the sequence of tokens $\text{Seq}_0 \in S_C$:

$$Seq_0 = \text{start}, a_1, b_2, b_3, c_4, \text{ end}$$

and the default seed points for Seq_0 given by the above mappings, shown in Fig. 6a (start and end tokens are omitted).

For $S \in P(Se)$, and $p \in$ {left, right, top, bottom, left_v, right_v, top_v}, we define the composition functions cut_P, add_P and null

$$\text{cut}_P(S) = S - \{p\}$$

$$\text{add}_P(S) = S \cup \{p\}$$

$$\text{null}(S) = S$$

and the transitions:

$$\text{A} \rightarrow \text{B} = (\{\text{cut}_{\text{bottom}}\}, \{\text{null}\})$$

$$\text{B} \rightarrow \text{B} = (\{\text{cut}_{\text{right\_v}}\}, \{\text{cut}_{\text{left\_v}}\})$$

$$\text{B} \rightarrow \text{C} = (\{\text{null}\}, \{\text{null}\})$$

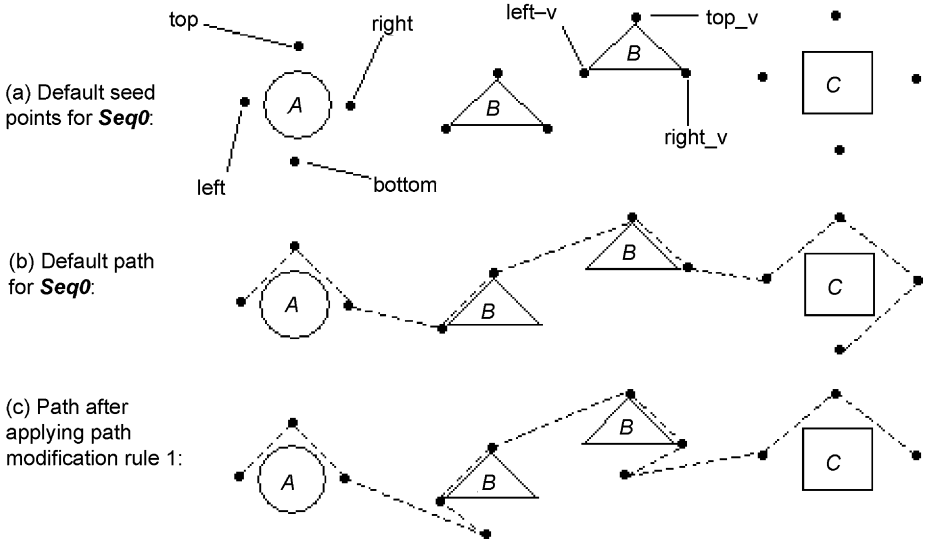


Fig. 6. Seed points and paths traversed prior to and after application of pm rule.

For Seq_0, we apply the transitions left to right cumulatively, i.e. A → B to $(m_a(a_1), m_b(b_2))$, obtaining $(m_a(a_1)', m_b(b_2)')$. Then B → B to $(m_b(b_2)', m_b(b_3))$, obtaining $(m_b(b_2)'', m_b(b_3)')$, and finally B → C to $(\text{m}_b(b_3)', m_c(c_4))$. We obtain the following seed points for each object:

$$\text{cut}_{\text{bottom}}(m_a(a_1)) = \{\text{left, top, right}\}$$

$$\text{cut}_{\text{right\_v}}(\text{null}(m_b(b_2))) = \{\text{top\_v, left\_v}\}$$

$$\text{null}\,(\text{cut}_{\text{left\_v}}(m_b(b_3))) = \{\text{top\_v, right\_v}\}$$

$$\text{null}\,(m_c(c_4)) = \{\text{left, top, right, bottom}\}$$

so establishing the path shown in Fig. 6b.

Having established how a default path is generated, we now informally describe application of pm rules, in terms of their Prolog formulation. A pm rule can be defined as an instance of the following Prolog schema:

p_m(Name_of_rule, Application_type, List_of_defining_subsequences)

Given a sequence of objects Seq, then the pm rule, name_of_rule applies to Seq if Seq is composed from the subsequences in List_of_defining_subsequences. Application_type is either single or multiple, denoting the frequency with which the rule is to be applied in Seq.

Consider the pm rule, rule 1, stating that given a sequence which begins with an object of type *a* and ending with an object of type *c*, then: (1) for any intervening pair of objects of type *b*, a bottom point should be added to each *b* type object; (2) the bottom point of the terminating *c* object should be removed from the path. This is expressed as follows:

$$\text{p\_m}(\text{rule 1, multiple,}[(\text{all}, \_, a\_\text{start}, \_), (\text{all}, [([b], [b], r1\_bb)], \_, \_),$$
$$(\text{all}, [([a, b, c], [c], r1\_c)], \_, c\_\text{end})])$$

List_of_defining_subsequences is a list of four-tuples of the form:

(Subsequence_name, Modifier_list, Subsequence_start, Subsequence_end)

where:

- Subsequence_name is a defined term mapping to a set of types. Intuitively, it declares that the parsed subsequence should contain a zero or greater iteration of objects of the stated type, e.g. referring to rule 1 above, the term ''all'' is mapped to all types a, b and c.
- Modifier_list is a list of three-tuples (L_1, L_2, Mod), where if $\tau_1 \in L_1$ and $\tau_2 \in L_2$ then Mod modifies the transition $\tau_1 \rightarrow \tau_2$ obtaining the modified transition $\tau_1 \rightarrow \tau_2$[Mod]. Hence, if there is a pair of neighbouring tokens X, Y of types τ_1 and τ_2, respectively, in the subsequence subsequence_name, then Mod modifies the transition from X to Y, e.g. ([b], [b], $r1\_bb$) denotes that $r1\_bb$ modifies the transition B $\rightarrow$ B between tokens b_i and b_j.
- As implied, subsequence_start and subsequence_end are terms denoting beginning and end subsequences (defined separately) of the subsequence subsequence_name. For example, the first defining subsequence identified in rule 1, is an arbitrary sequence of tokens, which begins with a_start, where a_start is defined separately as the pair start, $a(i)$ in the Prolog fact bound(a_start, [start, $a(i)$]).

A ''meta-interpreter'' has been written in Prolog, for translating each pm rule description to a set of grammar rules. Thus, one accounts for the applicability of pm rules by successively parsing a sequence of tokens with each pm rule, where each successful parse results in appending modifiers to each transition. The modified transitions are separately defined.

Let Seq_0 be the sequence (start, a_1, b_2, b_3, c_4, end). This is parsed by the translated rule 1, obtaining Seq_1 with the transitions A $\rightarrow$ B, B $\rightarrow$ B[$r1\_bb$], B $\rightarrow$ C[$r1\_c$] (if in addition we had the pm rules, rule 2,. . ., rule n, then Seq_1 would subsequently be parsed by the translated rule 2 obtaining Seq_2, and so on, until the final Seq_n), where:

$$\text{B} \rightarrow \text{B}[r1\_bb] = (\{\text{cut}_{\text{right\_v}}, \text{add}_{\text{bottom}}\}, \{\text{cut}_{\text{left\_v}}, \text{add}_{\text{bottom}}\})$$

$$\text{B} \rightarrow \text{C}[r1\_c] = (\{\text{null}\}, \{\text{cut}_{\text{bottom}}\})$$

Applying the transitions left to right cumulatively on Seq_1 now yields:

$$\text{cut}_{\text{bottom}}(m_a(a_1)) = \{\text{left, top, right}\}$$

$$\text{add}_{\text{bottom}}(\text{cut}_{\text{right\_v}}(\text{null}(m_b(b_2)))) = \{\text{bottom, left\_v, top\_v}\}$$

$$\text{null}(\text{add}_{\text{bottom}}(\text{cut}_{\text{left\_v}}(m(b_3)))) = \{\text{top\_v, right\_v, bottom}\}$$

$$\text{cut}_{\text{bottom}}(m(c_4)) = \{\text{left, top, right}\}$$

and the modified path shown in Fig. 6c is now generated as a result of applying the example pm rule.

3. Results

3.1. Active shape models image interpretation

The mean shapes that are produced for the upper and lower dentition are shown in Fig. 7. Fifty-four examples were used for the upper dentition, 61 for the lower. The eights, or

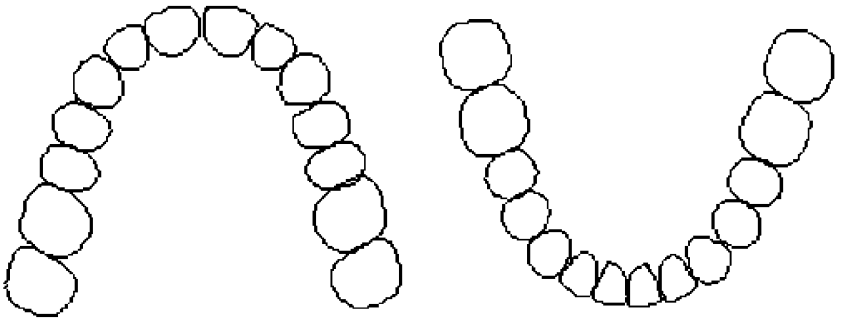


Fig. 7. The mean upper and lower dentition.

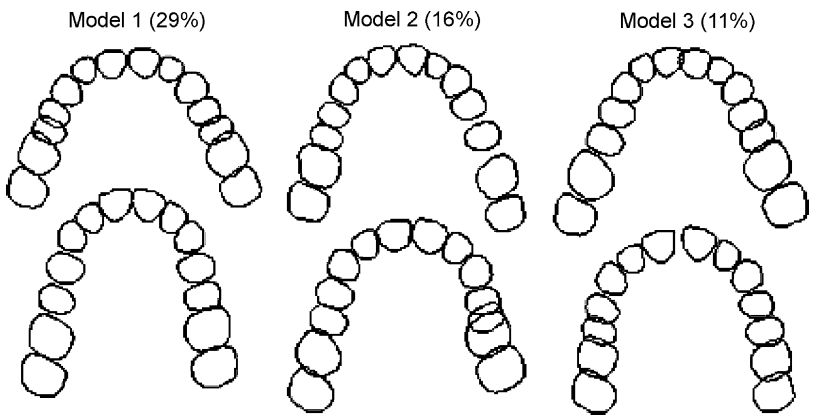


Fig. 8. The first three modes of variation for the upper arch. Each column shows a different mode, with the two rows showing the variation captured by each. Similar modes can be determined for the lower arch.

terminal molars, are not included in the model because they are present in so few of the cast models in the training set. These synthesised shapes correspond closely to the idea of the 'normal' dentition. Fig. 8 shows three modes of shape variation produced by the PCA for the upper dentition. The first mode (first column) has captured a change in arch width from wide to narrow, accounting for about 30% of the variation that is seen. The second mode consists of tooth movements, with very little change in the arch shape itself. The third mode contains changes in arch shape between square and triangular. The model shows teeth overlapping. This reflects the presence of partially edentulous patients in the training set and is a direct consequence of the method for dealing with the missing data by using patching-in techniques. This is not a problem because when fitting to a new example the user indicates (either manually or automatically via a patient record system) which teeth are missing before any attempt to fit the dentition, so the algorithm only searches for teeth that are present. In both cases, over half the variation in shape seen across the training set can be accounted for a linear sum of the first three modes (ordered by magnitude). Twenty modes in total explain 95% of the variation; with 20 parameters, every dentition—seen and

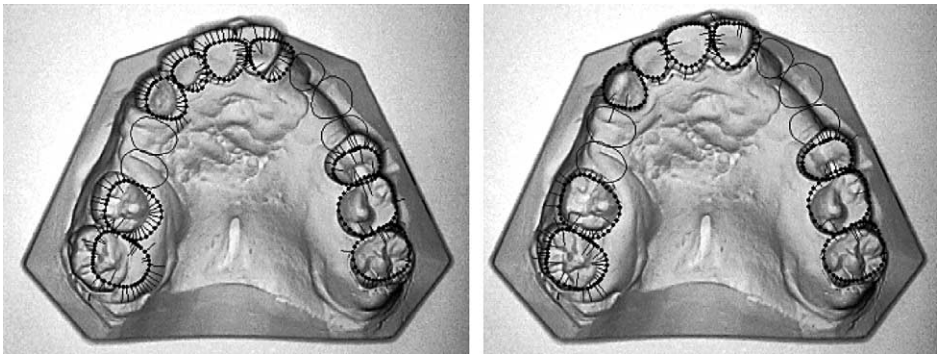


Fig. 9. Fitting the template to an image: initial placement of the template (left) and final fit (right). The missing teeth are shown but do not drive the fitting process.

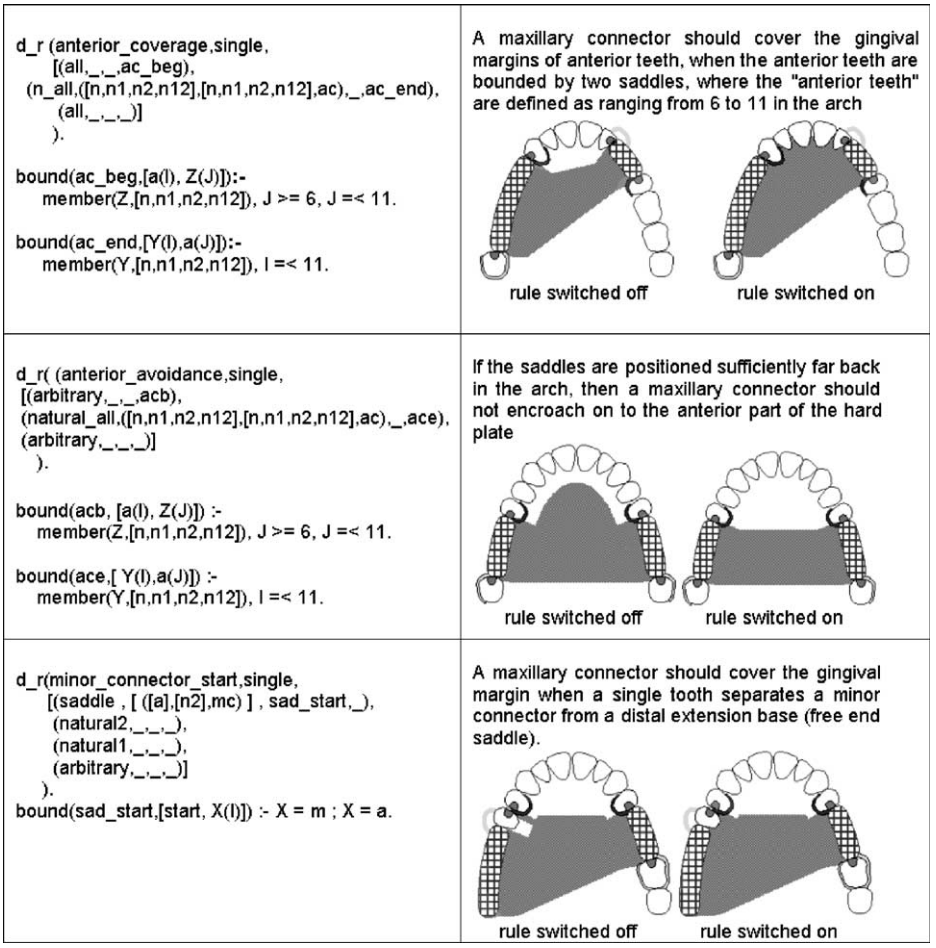


Fig. 10. Automated generation of major connector outlines, with rules switched off and on.

unseen—that is consistent with the training set can be described to a certain degree of accuracy. These 20 parameters could be used to classify dentitions or to compare variations between different populations. Here, we use them as described to search for structure in images.

Fig. 9 shows an example of the fitting procedure on an image. The template moves, rotates, scales and deforms to match itself to the image. In the current implementation the process typically takes under a minute on a 400 MHz PentiumIII. The final configuration of the template yields the outlines for the teeth that are used in RaPiD.

3.2. Improved RPD design in RaPiD

We have made use of path grammars to automate the generation of the outline of the major connector. Seed points for each tooth type in Table 1 have been defined. Path

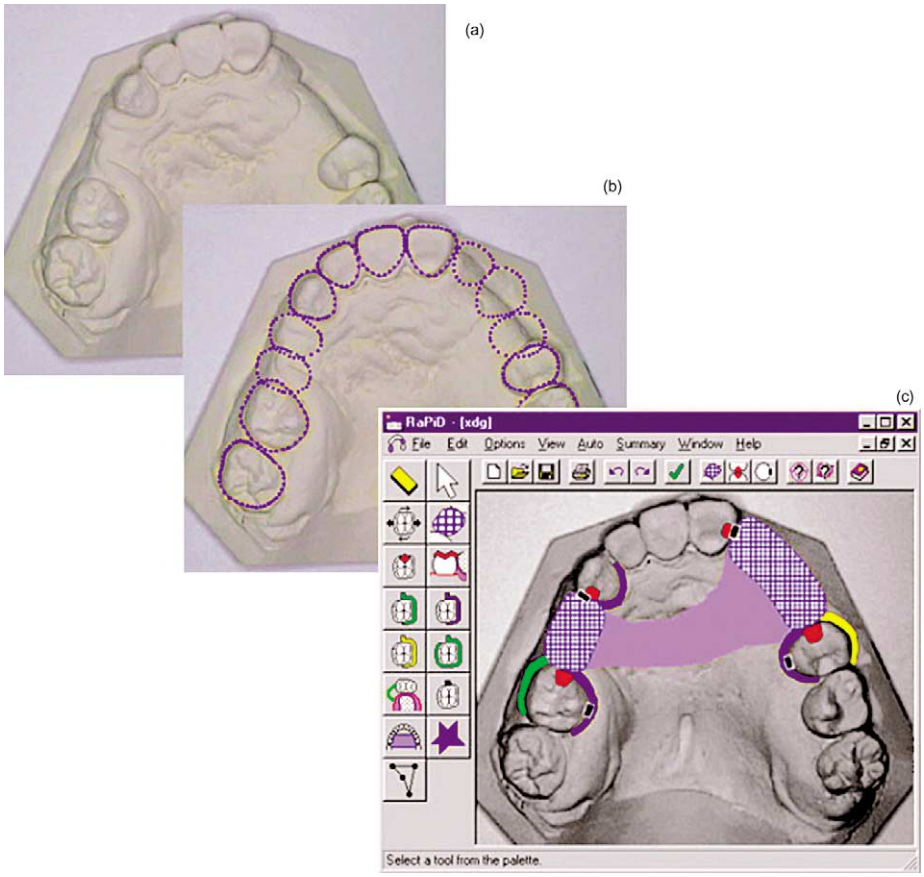


Fig. 11. An illustration of the combined process. An occlusal photograph of the cast (a) is automatically landmarked to find the outlines of the teeth (b) and then those outlines are used to automatically create a customised design (c).

modification rules corresponding to design rules, and transitions defined for each type of tooth to tooth transition have also been encoded in Prolog. A sequence of teeth is then successively parsed by each rule. Fig. 10 shows three of these design rules, the third of which is a specification of the connector rule described in Section 2.2.1. The rule applies if the token sequence can be described by a saddle subsequence, where saddle maps to a possibly empty subsequence of missing teeth, followed by artificial teeth, and sad_start specifies that the first two tokens are start followed by *m* or *a*. Any transition *a-n*2 in (actually terminating) the saddle subsequence is modified by mc. This modifier modifies a transition resulting in the seed points A,F,G in Fig. 5d rather than A,. . .,G in Fig. 5c.

Notice that it is relatively simple to customise use of the path grammar to switch design rules on and off. For each design rule shown, Fig. 10 shows the major connector outline obtained with the rule switched on and off. Also, note that the user may make use of a tool to make minor refinements to the automatically generated outlines (such as smoothing out sharp edges) or indeed make more major adjustments.

3.3. Combined result: customised, automated design

Finally, in Fig. 11 we show the complete process, from a photograph to a customised design. The details of the template fitting can be hidden from the user, who will only interact with the final stage (c)—the RaPiD software.

4. Summary and conclusions

This paper has described a second research phase in the RaPiD project, involving the customisation and automation of the design of dental prostheses. In addition to the built-in expert design rules, we have demonstrated the need for and use of a combination of different abstractions within a single knowledge-based design system: modes of shape variation, a logic database describing design components and their inter-relationships, a declarative description of graphical components, and a path grammar for connecting components. Thus, the RaPiD system now contains a rich combination of symbolic and numeric models, and the design process is almost entirely automated. Other approaches to this design problem have either been restricted to simple computer-assisted learning techniques [1] or to preformed components with little or no scope for customised design [14,18].

Designing an RPD on top of a clinical photograph of the patient's teeth means that the dentist has a wealth of relevant clinical information immediately to hand to assist in the design process. This represents a considerable advance over the present practice of designing on a standardised dental arch diagram which is devoid of such information. The detailed anatomical information provided by the photograph will allow a much more precise positioning, shaping and sizing of RPD components. It will also assist in the accurate interpretation of the design diagram by the dental technician, thus reducing the chance of errors occurring in the production of the RPD framework.

Note that, occasionally, the fitting of a dentition template to an unseen image does not work. Because of the uniformity of colour across the cast and the presence of spurious

ridges and folds, the template fitting can diverge from the desired teeth. Therefore, before such a system could be widely accepted, its robustness would have to be properly evaluated on a range of cast images. In addition, the reliability of the fitting might be improved by the adoption of active appearance models [2].

Having a method for automatically extracting the dentition of a patient from a photograph of a cast of their teeth is of benefit for purposes other than RPD design. The tooth outlines could be stored along with the image to form an exact orthodontic record of the patient's dentition at a specific point in time. The outcome of treatment could be evaluated quantitatively by direct comparison of the dentition before and after. The dentition could also be used for studies showing changes over time, since there is no associated health risk with taking an impression for a cast, unlike with cephalometrics for example where radiographs are required. A fourth use is for biometric studies of variation within and between populations, based on the full shape of the individual dentitions.

Automating generation of an RPD's major connector enhances the RPD design process in RaPiD. The user is no longer required to design this component manually; a process that proved to be both time-consuming and frustrating. Furthermore, the connector is now designed so as to conform to design constraints. Initial attempts to automate connector generation made use of logic grammars. This approach suffered from a number of inefficiencies which were overcome by the more generic path grammars. The latter approach also allowed for users to selectively apply design rules.

Our development of path grammars for path determination subject to design constraints, represents a novel application of logic grammars. The approach is generic enough that it may be applied in other domains, as suggested in Section 2.2.2 by the ''toy'' example of a robot determining a path. It remains to apply this work to other domains to further justify our claims.

In conclusion, the advances described in this paper will ensure that RaPiD can continue to play a role for general dental practitioners in assisting with and encouraging better design. The next obvious step in improving computer-aided RPD design is to consider the 3D aspect of the components. The precise design of clasps, for example, must currently be completed manually because it requires the 3D shape of the tooth that it encircles. If a model of a patient's dentition is digitised in 3D it would be possible to design an RPD with in-place reference to the 3D surfaces generated. If sufficient accuracy can be achieved and the required materials were compatible, computer-aided manufacturing (CAM) might then be used to directly manufacture the RPD, for example using selective laser sintering. This could drastically reduce the time taken for manufacture of an RPD for a patient, as well as potentially save money. One approach to this, using laser lithography, failed to restrict errors in component thickness to a level that the human tongue can comfortably accommodate [15]. The recent introduction of small, accurate and relatively cheap 3D laser scanners means that fully automated 3D design of RPDs is now realisable.

References

[1] Beaumont AJ, Bianco HJ. Micro-computer removable partial denture design. J Prosthet Dent 1989;6(2):417–21.

[2] Cootes TF, Edwards GJ, Taylor CJ. Active appearance models. In: Burkhardt H, Neumann B, editors. Proceedings of the European Conference on Computer Vision 2. Berlin (Freiburg, Germany): Springer, 1998. p. 484–98.
[3] Cootes TF, Taylor CJ, Cooper DH, Graham J. Active shape models—their training and application. Comput Vis Image Understand 1995;61:38–59.
[4] Davenport JC, Hammond P. The acquisition and validation of removable partial denture design knowledge. Part I. Methodology and overview. J Oral Rehabil 1996;23(3):152–7.
[5] Davenport JC, Hammond P, de Mattos MG. The acquisition and validation of removable partial denture design knowledge. Part II. Design rules and expert reaction. J Oral Rehabil 1996;23(12):811–24.
[6] Davenport JC, Basker RM, Heath JR, Ralph JP, Glantz O, Hammond P. A Clinical Guide to Removable Partial Denture Design. London: BDJ Books, 2000.
[7] EC Medical Devices Directive no. 10. Guidance notes for manufacturers of dental appliances (custom-made devices). Medical devices agency. Department of Health. London WC1B 5EP.
[8] Goodall C. Procrustes methods in the statistical analysis of shape. J R Stat Soc B 1991;53:285–339.
[9] Gower J. Generalized procrustes analysis. Psychometrika 1975;40:33–51.
[10] Hammond P, Davenport JC, Fitzpatrick FJ. Logic-based integrity constraints and the design of dental prostheses. Artif Intell Med 1993;5:431–66.
[11] Hammond P, Wells P, Davenport JC. DECADE: declarative computer-aided design environment, in: INAP-98. Proceedings of the International Conference on Applications of Prolog, 1998, Tokyo, Japan.
[12] Hutton TJ, Cunningham S, Hammond P. An evaluation of active shape models for the automatic identification of cephalometric landmarks. Eur J Orthod 2000;22(5):499–508.
[13] Hutton TJ, Hammond P, Davenport JC. Active shape models for customised prosthesis design. In: Horn W et al., editors. Proceedings of the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making, Lecture Notes in Artificial Intelligence 1620. Berlin (Aalborg, Denmark): Springer, 1999. p. 448–52.
[14] Maeda Y, Tsutumi S, Minoura M, Okada M, Nokubi T, Okuno Y. An expert system for designing removable partial dentures—preliminary report. J Osaka Univ Dental School 1985;25:79–84.
[15] Maeda Y, Minoura M, Tsutumi S, Okada M, Nokubi T. A CAD/CAM system for removable partial dentures. Part 1. Fabrication of complete dentures. Int J Prosthod 1994;7(1):18–21.
[16] Nilsson U, Matuszynski J. Logic, Programming and Prolog. New York: Wiley, 1990.
[17] Stilwell C. Designs on Prosthetics. Dentist 2001;60–8.
[18] Wicks RR, Pennell ME. Computer-assisted design guide for removable partial denture frameworks. Trends Tech Contemp Dent Lab 1990;7:51–3.