Rapport d'activité mi-parcours : TriComp

4 Novembre 2014

William Aufort

Julien Bensmail
coordinateur

Agathe Herrou
chef de projet

Romain Labolle

Frédéric Lang

Maxime Lesourd

Laureline Pinault

Léo Stéfanesco

Résumé

Ce document présente le rapport d'activité mi-parcours de notre projet TriComp. Y sont détaillés le travail fourni jusqu'à présent dans les différents groupes de travail ainsi que les diverses modifications qui ont été effectuées par rapport à la proposition de projet.

Table des matières

1 Changement par rapport à la proposition

1.1 Niveau de difficulté des tricots

Durant les premières semaines nous avons effectué une distinction entre le tricot plat et le tricot cylindrique (celui-ci nécessite plus d'aiguilles) qui permettait de créer des objets "cylindriques" (manches, chaussettes...) sans utiliser de coutures. Pour raison de simplicité, nous avons préféré nous concentrer sur le tricot plat, et donc sur l'assemblage de pièces par coutures.

1.2 Représentation intermédiaire

Il avait initialement été prévu de développer une représentation intermédiaire, où le tissu serait représenté sous la forme d'un graphe, dans lequel les sommets correspondraient aux mailles et les arêtes à la manière dont les mailles interagissent. Cette représentation avait pour but primaire d'aider à la détection de configurations impossibles. Cependant, après avoir approfondi les langages de bas et haut niveaux, nous nous sommes rendus compte que ce formalisme n'était pas intrinsèquement nécessaire. Nous avons donc abandonné l'utilisation globale de cette formalisation, pour ne se concentrer que sur une étude de motifs impossibles ponctuelle et non systématique.

1.3 Calendrier

Nous détaillons ici les changements qui ont été effectués par rapport au calendrier fourni dans la proposition de projet.

1.3.1 Interface graphique

Nous avions prévu d'avoir une interface graphique fonctionnelle dès lors que la première version du langage aurait été définie et compilable. Cependant, elle s'est révélée plus difficile à mettre en place que ce que nous avions imaginé, notamment concernant l'affichage des tricots et l'intégration du compilateur et du traducteur en instructions utilisateur.

La deadline que nous nous étions fixée pour la version 1 du langage sera donc repoussée, et cela devient notre priorité.

1.3.2 Versions supérieures du langage

La définition des versions 2 et 3 du langage n'a pas été faite formellement, mais les différences avec la version 1 se réduisant à l'ajout de nouvelles opérations au langage (respectivement diminutions/augmentations et croisements d'éléments de tricot), une telle définition n'était pas forcément nécessaire en dehors de l'implémentation. Cependant, les problèmes que pourraient introduire ces versions ont été étudiés, notamment la question de la répartition des diminutions sur un trapèze afin que sa pente soit régulière, ou les incompatibilités que pourraient poser l'introduction des tresses.

2 Travail fourni

Nous exposons dans cette partie le travail fourni jusqu'à présent dans les différents groupes de travail. Nous rappelons que tout notre travail (logiciel, documents et site web) se trouve sur le dépôt Git du projet TriComp (https://github.com/TriComp/).

2.1 Définition des différents langages

L'étape cruciale de définition des langages a été effectuée. Nous exposons ici avec précision ces différents langages.

2.1.1 Instructions utilisateur

Une première version du langage des instructions utilisateur a été établie, avec des tests sur des exemples. Nous avons maintenant des outils suffisants pour expliquer comment tricoter un modèle simple, comme par exemple un tricot rectangulaire avec un motif régulier (c'est le cas pour une écharpe).

Concrètement, une description bas-niveau d'un tricot, quasiment identique aux instructions utilisateur, mais en langage machine, a été mise en place. Le tricot est ici vu comme un type OCaml :

```
type stitch     = Endroit | Envers | Torse | Vide
type row        = stitch list * int
type trapezoid = row list * int
type piece      = parallelogram list
```

Une maille (`stitch`) est décrite par son type : endroit, envers ou torse. Le type Vide est utilisé pour décrire les premières mailles montées (et sera également utilisé par la suite pour décrire des augmentations). Une ligne (`row`) est décrite par le motif minimal qui la décrit (sous forme d'une liste de mailles) et par un entier indiquant le nombre de fois que ce motif doit être répété. De la même manière un trapèze tricoté (`trapezoid`) est décrit par le motif minimal qui la décrit (sous forme d'une liste de lignes) et par un entier indiquant le nombre de fois que ce motif doit être répété. Enfin une pièce de tricot (`piece`) est décrite par une liste de trapèzes tricotés.

Depuis cette description, on peut obtenir les instructions utilisateur en français pour réaliser le tricot. Pour cela on utilise un traducteur que nous avons implémenté.

L'exemple suivant donne une liste d'instructions que l'on peut obtenir avec le traducteur :

Exemple 1. *Suivez les instructions suivantes pour obtenir votre tricot :*
- *Rang 1 : Tricotez un point vide, et répétez 12 fois ce motif.*
- *Rang 2 : Tricotez un point endroit puis un point envers, et répétez 6 fois ce motif.*
- *Rang 3 : Tricotez un point envers puis un point endroit puis un point envers, et répétez 4 fois ce motif.*
- *Rang 4 : Tricotez un point envers puis un point endroit, et répétez 3 fois ce motif. Tricotez un point endroit puis un point envers, et répétez 3 fois ce motif.*

Tricotez au total 10 fois ces 3 derniers rangs.

Par ailleurs nous avons réfléchi à la forme que nous souhaiterions donner aux instructions utilisateur pour des tricots plus complexes, comme par exemple des tricots nécessitant plusieurs pièces tricotées indépendamment reliées par des coutures. Le format de sortie souhaité est détaillé ci-dessous :

1. **Titre du tricot**
   On présente d'abord quelques informations préliminaires : matériel requis (aiguilles, laine...), auteur du modèle...
2. **Présentation du patron du tricot**
   Liste des différentes parties à tricoter, dans l'ordre où elles doivent être tricotées.

3. **n-ième pièce à tricoter**
   Instructions pour tricoter la n-ième pièce
   Un exemple :
   – aiguilles et laine à utiliser
   – lancer le tricot avec 21 boucles sur une première aiguille (ligne 0)
   – instructions :
     ⋆ 1ere ligne : un point endroit, un point envers, un point endroit, puis répéter ce motif 6 fois (jusqu'au bout de la ligne)
     ⋆ 2e ligne : un point envers, un point envers, un point endroit, puis répéter ce motif 6 fois (jusqu'au bout de la ligne)
     ⋆ Répéter ces 2 lignes 10 fois pour obtenir 22 lignes
     ⋆ 23e ligne : un point envers, un point endroit, puis répéter ce motif 4 fois. Ensuite faire une diminution, et encore répéter le motif 5 fois.
4. **Keme méta-instruction**
   Instructions d'assemblages
   Un exemple :
   - Prendre la partie I.
   - Coudre son bord gauche avec son bord droit.

2.1.2 Langage descriptif

Le langage descriptif permet d'interfacer le traducteur et l'interface graphique. Les contraintes étaient d'avoir un format de description facilement manipulable à travers l'interface graphique tout en gardant la structure d'un tricot. La solution adoptée est de voir un tricot comme un assemblage d'éléments qui sont des structures arborescentes de trapèzes tricotés. Ces éléments sont appelés *pièces* dans notre langage. Afin de former le tricot final, certaine pièces tricotées l'une à la suite de l'autre sans rupture apparente pour l'utilisateur (le lien se fait par un connecteur inclut dans la description de la première pièce à tricoter dans l'ordre), tandis que d'autres pièces seront assemblés par couture, ces instructions étant transmises quasiment sans modification à l'utilisateur (elles ne sont donc pas traitées lors de la compilation).

Nous avons décidé d'utiliser les trapèzes comme forme de base pour plusieurs raisons. Tout d'abord, le tricot s'effectue par rangs (lignes) successives et parallèles (car tricotées les unes sur les autres). De plus, même si l'on veut faire des bords non droits, comme le tricot est discret, on peut se ramener par approximation à une suite de trapèzes. Des macros seront implémentées par la suite afin de permettre dans le langage de définir la courbe que l'on veut sans avoir à écrire toute la suite de trapèze.

Les trapèzes sont tricotés selon divers motifs (points si on suit le vocabulaire du tricoteur). Pour le moment ces motifs sont fixés (jersey endroit, jersey envers, mousse), mais par la suite le langage intégrera la possibilité de définir des points à base de mailles endroit et envers. Utiliser des trapèzes est selon nous intéressant, même lors de l'édition du tricot par l'utilisateur. En effet, chaque modification (ajout de points) correspondra à des séparations de trapèzes, ce qui est déjà géré par notre langage.

Plus précisément, le langage décrit les relations entre les trapèzes qui se touchent via leurs bases. Outre le cas où deux trapèzes partagent la même base, la base d'un trapèze peut être adjacente à plusieurs trapèzes : on décrit alors leur séparation via `"split"` et la réunion de deux trapèzes via `"link"`. Les bases correspondant au début et à la fin du tricot sont indiqués par les mots clés `"start"` et `"stop"`.

Exemple 2. *Un code du type de celui-ci décrit un poncho, illustré par la figure 1.*

```
piece my_piece := start size
                  || trapezoid (parametres)
                  || split { trapezoid (...)
```

```
                        || link left next
                        }{ trapezoid (...)
                        || link right next
                        }

piece next := start size
            || trapezoid (...)
            || stop
```

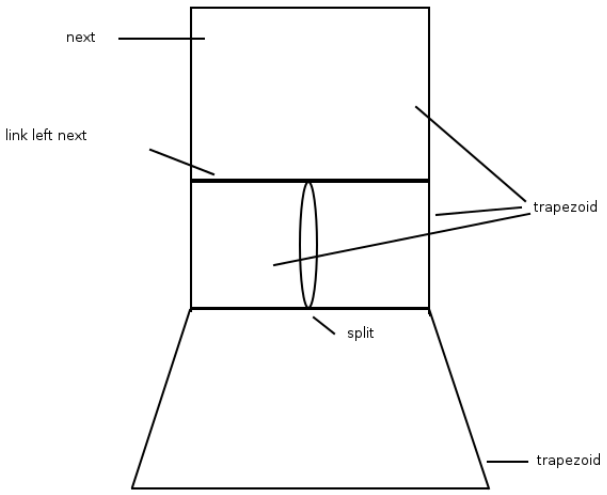

FIGURE 1 – Le poncho généré par le code précédent

Dans cet exemple, les points de suspension contiennent les paramètres des trapèzes (taille des bases, hauteur, décalages et type de point). Le schéma 2 illustre ces informations sur un trapèze.

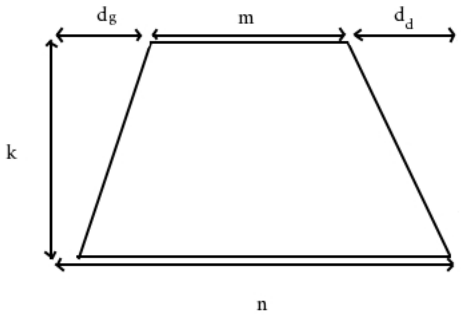

FIGURE 2 – Les paramètres d'un trapèze que l'on utilise dans la description

Cette syntaxe, qui permet déjà de travailler avec les augmentations et diminutions (ce qui n'était prévu que pour la version 2), a été définie et le parser utilisé par le compilateur a été implémenté.

En outre la réflexion sur la manière d'intégrer les tresses (version 3) à ce langage a été entamée.

2.2 Compilateur

Contrairement à nos prévisions le compilateur n'est pas encore fonctionnel. Ceci est dû à l'abandon de la représentation intermédiaire sur laquelle nous avions travaillé au début.

Pour l'instant seul le parser du langage descriptif a été implémenté. Cependant l'algorithme pour transformer les tricots possibles avec la version 1 de notre logiciel (des trapèzes) est défini et il ne reste plus qu'à l'implémenter. Le travail avec des trapèzes est à l'étude, nous nous penchons actuellement sur deux algorithmes afin de distinguer les avantages et inconvénients de chacun par rapport à notre objectif. De plus, le travail de réflexion sur la compilation lorsqu'il y aura des tresses a déjà commencé.

2.3 Module de vérification

Bien qu'il ait été prévu à l'origine de s'intéresser à ces questions que tard dans le projet, la question de l'abandon de la représentation intermédiaire nous a contraints à nous pencher dès maintenant sur ce module.

Ce module de vérification devait réaliser deux fonctions distinctes :

1. Sa première fonction à l'origine était de pouvoir détecter des impossibilités s'il y avait superposition de tresses avec des points incompatibles (par exemple des augmentations ou des diminutions). Il a été décidé de ne pas laisser le choix à l'utilisateur de où placer ses augmentations et ses diminutions mais de les placer dans une lisière (procédé couramment utilisé par les tricoteurs). Ainsi il n'y aura pas de problèmes de ce côté. De plus les tresses seront une décoration décrite "au-dessus" des points, au niveau du langage descriptif, elle remplacera donc les motif décrits "en dessous" par l'utilisateur.
2. Une deuxième fonction a été ajoutée : vérifier que le nombre de mailles tricotées à la ligne $i+1$ est le même que le nombre de mailles que l'on obtient après avoir tricoté la ligne i. Cette vérification, simple à implémenter, va permettre par exemple de vérifier que les points qu'aura défini l'utilisateur sont cohérents.

2.4 Interface graphique

Nous détaillons dans cette section les fonctionnalités implémentées jusqu'à présent au niveau de l'interface graphique.

2.4.1 Squelette de l'interface / Aspect

La première étape du développement de l'interface a été la mise en place de son squelette. Plus précisément, nous avons défini l'aspect général de l'interface ainsi que les différents outils que nous souhaitons mettre à disposition. Ces outils sont représentés à l'aide de boutons ou d'options encore non fonctionelles pour la plupart. On distingue notamment :

- Les options relatives au logiciel TriComp (choisir un point, faire une tresse...). Celles-ci peuvent être sélectionnés grâce à des boutons dont l'aspect n'est pas encore fixé, car leur implémentation se fera au fur et à mesure et en fonction de l'avancement global du projet.
- Les options que l'on trouve dans tout logiciel (ouvrir, sauvegarder, quitter, ...). Ces options sont fonctionnelles (ou bientôt fonctionnelles) car les formats de données pour la sauvegarde ont été définis. Nous travaillons directement sur le format de fichier associé au langage descriptif. Ce type de fichier portera l'extension `.tricot`.

La figure 3 présente l'interface utilisateur du logiciel. Celle-ci se décompose en trois parties :

1. un panneau qui contiendra les différents outils de tricot mis à la disposition de l'utilisateur.
2. une fenêtre d'affichage du tricot ;
3. une fenêtre qui contiendra la liste des instructions générées par l'utilisateur.

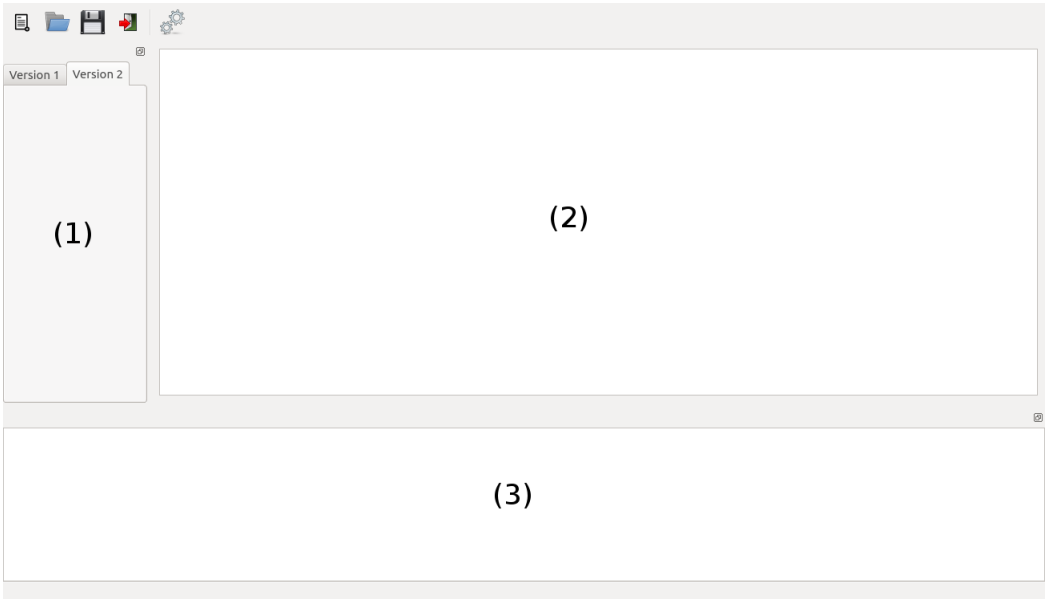

FIGURE 3 – L'aspect général de la fenêtre principale, avec les trois parties principales mises en évidence

2.4.2 Affichage des tricots

L'affichage des tricots en lui-même est en cours d'implémentation. Il est prévu de les afficher sous forme de patron, avec éventuellement la possibilité d'afficher la progression d'un tricot en cours de réalisation. Nous travaillons avec des classes de Qt qui permettent la gestion et l'affichage d'objets 2D (`QtGraphicsScene` et `QtGraphicsItem` par exemple).

Nous avons travaillons également sur un parser pour permettre l'utilisation du fichier `.tricot` par l'interface graphique. Celui-ci est sensiblement identique à celui implémenté en Ocaml. Il est en cours de finition (notamment son incorporation dans le projet Qt).

2.5 Site Web

Un site web a été déployé à l'adresse `http://tricomp.github.io`, il est hébergé sur Github (utilisé comme serveur Git du projet). Le site est basé sur Jekyll, un CMS en Ruby spécialisé dans les blogs et qui a la particularité de ne pas utiliser de base de données, avec l'avantage que Github gère Jekyll automatiquement. Le site est notamment une vitrine pour le projet : il en contient une présentation rapide, avec des liens vers le code source sur Github. À terme, le site sera aussi un support pour télécharger et installer le logiciel TriComp.

De plus, le site contient pour l'instant un article sur ce qui existe déjà sur Internet autour du tricot (notamment certains projets proches de TriComp qui ont été détaillés dans la proposition de projet). D'autres articles devraient s'y rajouter, ainsi qu'une page pour aiguiller rapidement l'internaute anglophone.

3 Conclusion

Malgré le travail fourni jusqu'à présent par toute l'équipe, tant du point de vue de la réflexion autour des différents langages que du code, la durée nécessaire à la mise en place d'une version entièrement fonctionnelle du logiciel a été sous-estimée. Néanmoins, la durée nécessaire pour les nouvelles versions

peut être revue à la baisse, car nos réflexions sur les différents aspects du projet ont permis d'avoir assez tôt une idée des difficultés qui nous attendent par la suite.