Workshop on Material Appearance Modeling (2018)
H. Rushmeier and R. Klein (Editors)

Image-based Fitting of Procedural Yarn Models

A. Saalfeld[1] and F. Reibold[1] and C. Dachsbacher[1]

[1]Karlsruhe Institute of Technology

Abstract

While common in real life, rendering fiber and cloth accurately is challenging. Recent fiber-based, procedural rendering approaches proved to be able to capture a great amount of details of real yarn. However, the current automatic method of fitting the model parameters is expensive and inaccessible as it relies on micro CT scans of the reference yarn. The alternative is to have an artist fit the parameters by hand, which is impractical because of the large number of parameters.
We present a proof-of-concept for a purely image-based approach to fit the parameters of a procedural yarn model. Using gradient descent and pixel-based loss functions, we are able to extract a subset of the model parameters from rendered images with known parameters. The appearance of the fitted models is nearly indistinguishable from the reference images.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation

1. Introduction

Cloth and fabric play a huge role in our everyday lives. Clothing, blankets, towels, furniture and many more things consist (mostly) of these materials. It is therefore no surprise that they are of importance for the field of computer graphics as well, be it in the form of games, movies or other entertainment software, design and simulation of clothing and fashion or for marketing purposes.

The appearance of fabric and cloth is heavily influenced by fiber-level details. To replicate the diversity of different kinds of cloth, we need advanced models that are able to capture many different aspects of the material.

While these materials can be rendered using volumetric approaches, in recent years fiber-based approaches became a popular alternative. These approaches calculate exact fiber curves and realize them instead of using volume rendering approximations.

Replicating the appearance of real world cloth is challenging as the models are usually too complex to be fit by hand. Zhao et al. [ZLB16] proposed an automated fitting approach for their model, but it involves using data of a micro CT scan. These are not widely available and can be expensive to obtain.

We therefore studied the possibility of fitting the same model using only a reference image instead of a CT scan.

As a first step towards this, we used a gradient descent optimizer with loss functions that calculate the per-pixel difference between the reference image and a candidate rendering. This rather simple approach yields good results for a subset of the model parameters when using an image of yarn rendered with the same rendering pipeline as the reference.

In this future, we plan to expend upon this approach to be able to fit the parameters of our model using photos of real-world yarn.

1.1. Previous work

Jakob et al. [JAM*10] published one of the first papers explicitly focusing on cloth rendering. They proposed a micro-flake model to allow for a volumetric approach to capture the appearance of cloth.

Khungurn et al. [KSZ*15] proposed a method to fit volumetric models to real world yarn using micro CT scans. In their algorithm, they convert the density distributions of the CT scan to a fiber-based representation before converting it into a volumetric model.

Zhao et al. [ZLB16] proposed a procedural fiber-based yarn model. They also describe an approach to fit their model to a CT scan using the fiber representation of Khungurn et al. [KSZ*15].

While the model by Zhao et al. [ZLB16] was intended for offline rendering, Wu and Yuksel [WY17] implemented a slightly simplified version of the model that can be evaluated in real time rendering applications. We will go into more detail about their model and the real time implementation in the following sections.

2. The procedural yarn model

The procedural yarn model we used is based on Wu and Yuksel [WY17], which is in turn a modified version of the model by Zhao et al. [ZLB16].

DOI: 10.2312/mam.20181194

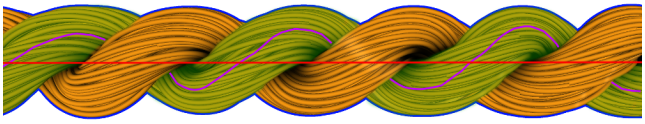

Figure 1: *This figure demonstrates the structure of yarn according to our procedural yarn model. Outlined in blue is the whole yarn, marked in green is one of the two plies in this rendering. The red and violet lines approximate the yarn and ply center curve respectively. The brownish tubes that make up the plies are the fibers.*

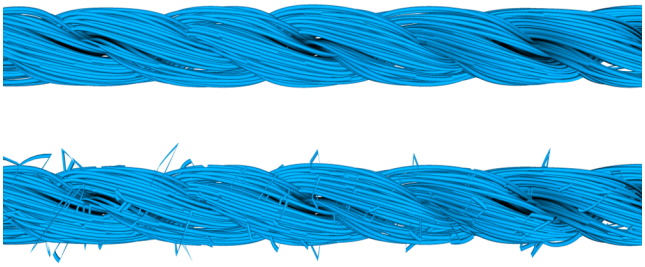

Figure 2: *Top: The rayon 2 yarn model of Zhao et al. [ZLB16], rendered with no flyaway fibers. Bottom: The same model with flyaway fibers.*

The model consists of three components: Fibers, plies and yarn. The yarn as a whole consists of (usually two to three) plies that have been twisted around a common (yarn) center. The plies consist of individual fibers, that have been twisted around the ply center. Plies can consist of up to hundreds of fibers [ZLB16]. This structure is visualized in figure 1.

Not all fibers are perfectly contained in their respective plies. Some of them stick out and contribute a lot to the overall appearance. Zhao et al. [ZLB16] coined the term fly-away fibers for them and distinguished between loop-type and hair-type fly-away fibers. A hair-type fly-away fiber is a loose end sticking out of the ply while a loop-type does not expose its ends - it essentially is a regular fiber that has been pulled out of the "flow" of the ply. See figure 2.

Our model uses 22 parameters to describe its structure. While it is mostly equal to the model by Wu and Yuksel [WY17], we used the approach of Zhao et al. [ZLB16] for hair-type fly-away fibers instead. This leads to half of our parameters describing the fly-away fiber density and curves, while the rest describes radii, twisting and number of plies and fibers as well as the fiber distribution and migration (periodic change in distance of a fiber to its respective ply center).

3. Real-time rendering

To render the procedural yarn model in real-time, Wu and Yuksel [WY17] proposed an approach that makes heavy use of the tessellation feature of modern OpenGL, more specifically the isoline tessellation.

This allows individual fibers to be generated on the fly and in parallel on the GPU. We only need to provide it with the model parameters and some additional data like the control points of the yarn curve and the position of (tessellation) patches along the length of the yarn.

The naive idea is to render each fiber using an isoline generated by the tessellation stage. Unfortunately current hardware does not support more than 64 isolines per rendering pass.

Wu and Yuksel [WY17] therefore proposed the use of so-called core fibers. These are thick fibers that follow the ply center and are textured to look like a collection of multiple individual fibers. To generate the necessary texture, the fiber curves are simplified in a way that the resulting texture can be tiled.

Using core fibers, the remaining isolines can be used to render the outermost fibers of each ply. While this approach can not fully capture the diversity in appearance of actually rendering all the individual fibers [LZB17], this approach can be good enough in many use cases.

4. Fitting the model

To fit the model, we make use of the real-time rendering approach by Wu and Yuksel [WY17]. It allows us to generate dozens to hundreds of close-up images of a thread of yarn per second. The core fiber approach is not useful for us as we generate only one image per model thus forcing us to regenerate the core fiber textures for every rendering anyway. Still, using isoline tessellation to render the yarn is a fast enough approach to allow us to fit most parameter combinations in under an hour.

We fit the model using an optimizer and loss functions that compare the images on pixel-level. We initially ran some tests with the Nelder Mead method (also known as downhill-simplex method) [NM65] and some variations of it. Unfortunately these tests have shown low accuracy and precision. We decided to settle on gradient descent with momentum [Qia99] as our optimizer, as it yielded promising results.

The loss functions we used compare pixels of a candidate rendering to the respective pixels in the reference image. We convert the pixels from RGB into HSV and compare two pixels by taking the absolute or the squared difference of their V component. We ignore the hue and saturation component because we do not fit the color and are able to mostly ignore it this way, similar to [AWW13].

We then add up these differences and divide by the number of non-background pixels in the reference image to get the loss value. The division normalizes the loss for the size of the thread of yarn both in resolution and relative to the size of the image.

5. Results

For certain parameters and paramter combinations our fitting approach works very well. Unfortunately this is not the case for all of them. Looking at figure 3 one can see why this might be: Parameters like r^{ply}, the radius of the plies, provide a meaningful change on the pixel level and therefore show a rather smooth gradient with a single minimum that can easily be found following the gradient

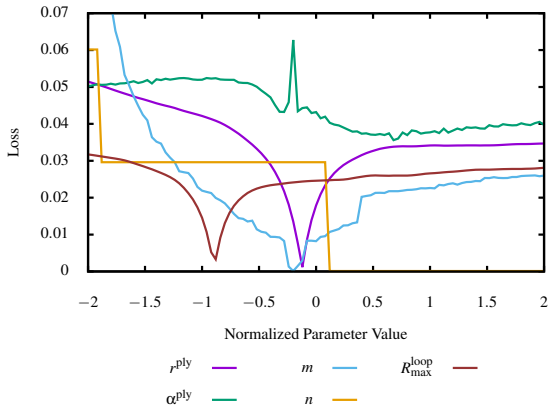

Figure 3: *Plots of the **absolute color distance** loss function. The parameters are normalized to the mean and standard deviation of the given parameter calculated from all models Zhao et al. [ZLB16] introduced. Every parameter except for the plotted one matches the parameters of their cotton 1 model and the loss is calculated against a rendering of the same yarn model.*

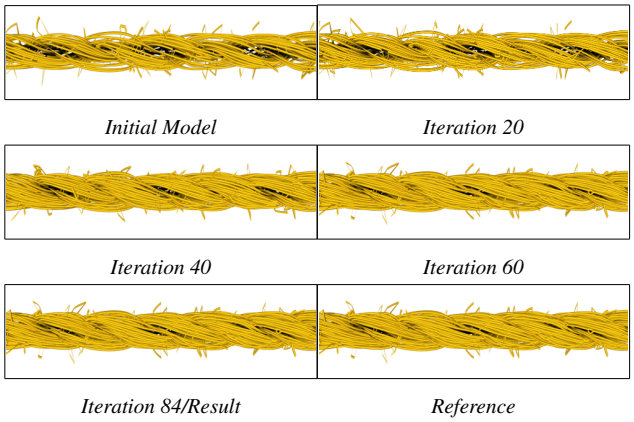

Figure 4: *Demonstration of the fitting process fitting R_{min} (the minimal distance between a fiber and its respective ply center) and m (the number of fibers per ply) to the polyester 1 model by Zhao et al. [ZLB16].*

curve. Meaningful in this case means that small changes in the parameter value, away from the optimal value, lead to small increases in loss.

Other parameters like α^{ply}, the parameter controlling the twist of the plies around the yarn center, do not provide these meaningful changes on the pixel level. They change pixels seemingly "at random". Therefore their gradient is bumpy and has a lot of local minima the optimizers can get stuck in even when reducing the problem to an 1D case.

While figure 3 only shows a reduced variation of the original fitting problem, the parameters that look like they could be fitted well on their own could in fact usually be fitted together with others parameters in this category. For most combinations we were able to find parameters with an average relative error below 0.1 in under an hour. Both fitting time and average relative error grew larger with higher dimensionality.

The decision between absolute or squared error as the used loss function had an influence on both fitting time and relative error. In most cases the absolute error loss function terminated slightly quicker but with a slightly higher relative error. However, we found exceptions to this rule.

While not directly comparable because of the vast difference in available data for fitting, we want to note, that Zhao et al. claim to be able to fit yarn in "several core hours" [ZLB16].

It is also noteworthy that our implementation of the real-time rendering approach by Wu and Yuksel [WY17] was significantly slower than their implementation according to their measurements. Depending on the test configurations our implementation took anywhere from circa 1.5 to over 100 times longer to render the images than their implementation.

6. Future work

Our fitting approach is not yet able to cross the domain gap to fit the yarn model to real world photos and even struggles with some of the model parameters given a reference image that has been created using the same rendering pipeline. We believe there are multiple ways to tackle this problem.

The loss functions currently work by comparing pixels of the candidate rendering to the respective pixels in the reference. This cannot work if the images were created with different methods.

Instead we could try to extract information about the yarn like the direction of fibers and plies directly from the reference image. We can then simplify the fitting problem: Although these directions depend on multiple parameters of our model, we can optimize only these to match the given estimated directions.

Counting fly-away fibers in a similar fashion to the fitting approach by Zhao et al. [ZLB16] might also be possible, although we only have 2D data instead of the 3D data they use. Still, we might be able to approximate the density of fly-away fibers directly from an image.

Alternatively it might be possible to augment the pure 2D image data with depth information to restore the surface of one side of the yarn. This could then be used to measure the fly-away fiber parameters.

Another approach to cross the domain gap involves machine learning and neural networks. Gaidon et al. [GWCV16] have shown that pre-training a neural network with virtual data and then fine tuning it to real world data can lead to highly precise and accurate results. This might also be applicable for fitting the procedural yarn model.

Image-based fitting might have other applications in cloth rendering as well. It might be possible to extract weaving patters from photos of woven cloth with similar ideas to the ones mentioned above. This would ease the digitization of real-world whole clothing even further.

7. Conclusion

We have shown that fitting procedural yarn models using only a reference image is possible for at least some parameters. For these parameters, the fitting process is reasonably fast and the results are visually nearly indistinguishable from the reference image.

We believe that our work is a proof of concept that image based fitting approaches for procedural yarn models can work. While our current methods were neither able to fit all parameters nor to cross the domain gap between virtual and real images, we propose ideas to improve upon our current approach.

References

[AWW13] AMANN J., WEBER B., WÜTHRICH C. A.: Using image quality assessment to test rendering algorithms. In *21st International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision in co-operation with EUROGRAPHICS Association, WSCG 2013, Plzen, Czech Republic, June 24-27, 2013* (2013), pp. 205–214. URL: http://wscg.zcu.cz/WSCG2013/!_2013-WSCG-Full-proceedings.pdf. 2

[GWCV16] GAIDON A., WANG Q., CABON Y., VIG E.: Virtual worlds as proxy for multi-object tracking analysis. *CoRR abs/1605.06457* (2016). URL: http://arxiv.org/abs/1605.06457, arXiv:1605.06457. 3

[JAM*10] JAKOB W., ARBREE A., MOON J. T., BALA K., MARSCHNER S.: A Radiative Transfer Framework for Rendering Materials with Anisotropic Structure. In *ACM SIGGRAPH 2010 Papers* (New York, NY, USA, 2010), SIGGRAPH '10, ACM, pp. 53:1–53:13. URL: http://doi.acm.org/10.1145/1833349.1778790, doi:10.1145/1833349.1778790. 1

[KSZ*15] KHUNGURN P., SCHROEDER D., ZHAO S., BALA K., MARSCHNER S.: Matching Real Fabrics with Micro-Appearance Models. *ACM Trans. Graph. 35*, 1 (Dec. 2015), 1:1–1:26. URL: http://doi.acm.org/10.1145/2818648, doi:10.1145/2818648. 1

[LZB17] LUAN F., ZHAO S., BALA K.: Fiber-Level On-the-Fly Procedural Textiles. *Comput. Graph. Forum 36*, 4 (July 2017), 123–135. URL: https://doi.org/10.1111/cgf.13230, doi:10.1111/cgf.13230. 2

[NM65] NELDER J. A., MEAD R.: A simplex method for function minimization. *The Computer Journal 7*, 4 (1965), 308–313. doi:10.1093/comjnl/7.4.308. 2

[Qia99] QIAN N.: On the momentum term in gradient descent learning algorithms. *Neural Networks 12*, 1 (1999), 145 – 151. doi:https://doi.org/10.1016/S0893-6080(98)00116-6. 2

[WY17] WU K., YUKSEL C.: Real-time Cloth Rendering with Fiber-level Detail. *IEEE Transactions on Visualization and Computer Graphics PP*, 99 (Aug. 2017), 1–1. doi:10.1109/TVCG.2017.2731949. 1, 2, 3

[ZLB16] ZHAO S., LUAN F., BALA K.: Fitting Procedural Yarn Models for Realistic Cloth Rendering. *ACM Trans. Graph. 35*, 4 (July 2016), 51:1–51:11. URL: http://doi.acm.org/10.1145/2897824.2925932, doi:10.1145/2897824.2925932. 1, 2, 3