Response to Miller's Critique on SEE++

This is a response to Joel M. Miller's critique on the SEE++ software system. The original text of this critique can be found on Miller's Website (**http://www.eidactics.com/Eidactics_Pages/Products/Orbit1.8/SEE-Critique.php**), alternatively there is a **local copy as PDF-file to download from this homepage**.

In this text we want to clarify that **Miller's critique is primarily based on an outdated version of the SEE++ software system**. Furthermore, Miller's critique strongly adopts the position that **all** model implementations of the SEE++ software system, especially the integration of the "Orbit™ 1.8" model in SEE++ are "flawed".
This response shall clearly show that Miller's critique is partly based on assumptions and misunderstandings.

Once more, we want to emphasize that we regard our work as continuation and prolongation of **Miller's innovative and scientifically eminently respectable work** in the field of biomechanical eye modelling. **It is no secret that SEE++ is partly based on ideas and concepts originally invented and implemented in the software system Orbit™ by Joel Miller (Eidactics).**

However, we still have the strong believe that Orbit™, as computer software (last updated version 1.8 from the year 1999), misses many things that make up a good usable product, and above all, without any doubt, Orbit™ is technically totally obsolete.

For us, the following citation from Miller's website perfectly characterizes the current technical status of the Orbit™ computer program:

"It had previously been possible to purchase a license …, however, we stopped selling licenses because: [1] we have not updated Orbit for Mac OS-X, and it must be run, somewhat clumsily, in "Classic mode" (Note: the new Intel Macs do not support Classic mode), [2] the "automatic parameter fitting" function does not work in Classic mode, and [3] "active pulleys" are not included in the model."

We also want to state that **SEE++ and the SEE-KID research project partially have overlapping objectives with Miller's work**, a fact that we are fully aware of and that **ideally could result in valuable cooperation on some common goals**.

In the following, we want to respond to Miller's statements on the SEE++ implementation:

1. ***"… some advanced aspects of the model are simple lookup tables, rather than proper biomechanical simulations."***
   It is indeed true that SEE++ uses measured data from experiments to simulate muscle forces and ocular counterroll. However, these data serve as reference for an average "normal" eye, **an approach that is identically implemented in Orbit™** and similarly used in SEE++ to predict ocular counterroll for average eyes. For simulating reference muscle forces, SEE++ uses **exactly the same muscle force data as Orbit™**, for simulating average "normal" ocular counterroll, SEE++ uses data from Schworm et al. (Orbit™ does not simulate counterroll).
   When finding fixing eye positions, SEE++ starts with a "healthy" reference eye. Additionally, the head tilt and respective counterroll angle is included as input parameter and the counterroll is added to the torsional value of the reference eye position. Then, innervations for each position are calculated iteratively until the fixing eye position matches a 2D fixation position. For fixation, only the abduction/adduction and elevation/depression can be chosen freely. This approach uses ocular counterroll data to specify torsional values for an average "normal" eye and thus affects

innervations for the fixing eye.

Besides muscle force and ocular counterroll simulations, SEE++ does not contain any other "(simple) lookup tables"!!!

2. ***"There is no escaping the fact that, in a eye with elastic pulleys or fat pads that allow globe translation, muscle forces affect muscle paths and, for the case in which one is solving for them, affect eye positions as well. These effects alter muscle path lengths, and so, muscle forces. Iteration (or something equivalent) is inescapable."***

This was a problem in the Version 5 of SEE++ (2005) and was corrected with SEE++ Version 6.1 in the beginning of 2006.

SEE++ calculates a side-slip angle based on muscle pulling direction, current muscle force, muscle length and location of muscle action circle. Therefore, the side-slip calculation from the Orbit™ model was adopted and a non-linear spring is used to model the relationship between muscle force and side-slip scaling. The "radial" force is calculated as that portion of the total force (muscle force in the direction of the pulley) that exerts force in the direction of the center of the current muscle action circle. It is important to note that for each new calculation for a given gaze position, side-slip and muscle force depend on each other. Thus, the side-slip calculation for a muscle in a specific gaze position ends up in an iterative calculation process, since a new gaze position enforces a new initial side-slip and a new muscle force.

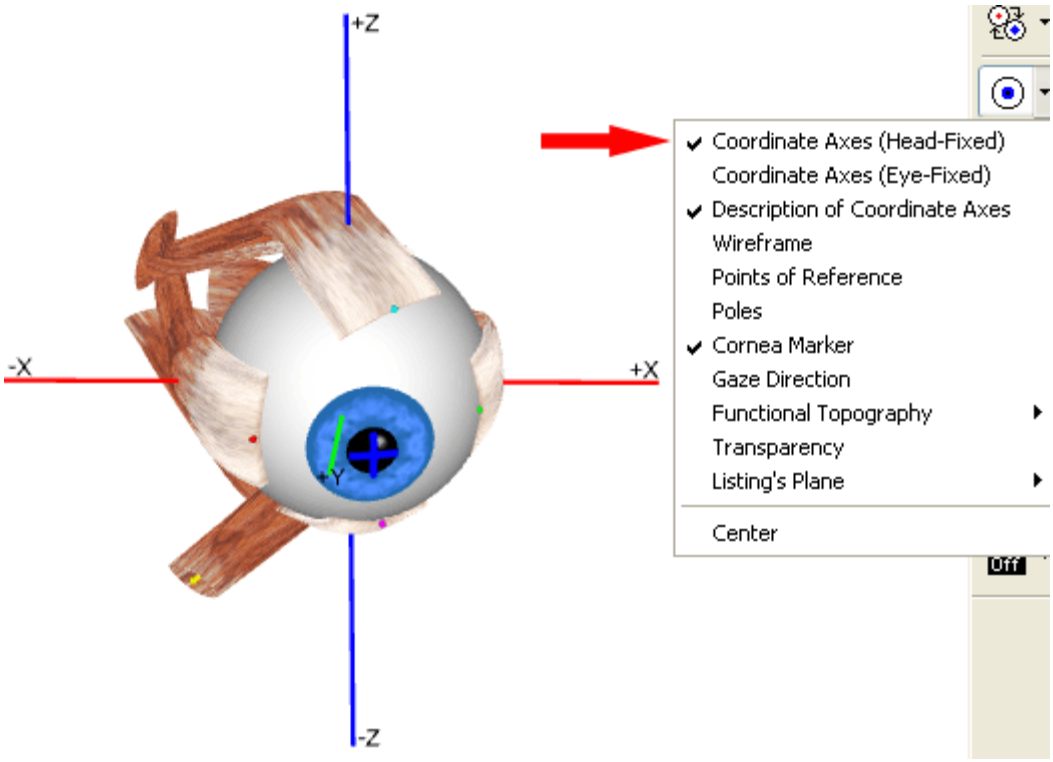
3. ***"The SEE++ implementation of "active pulleys" is simply an incorporation of Kono et-al's (2002) measurements as an internal lookup table. SEE++ does not implement a biomechanical model with movable pulleys. A proper model must, of course, represent the underlying mechanics, and then produce measurements such as Kono's, as an emergent property."***

SEE++ does NOT implement Kono et-al's (2002) measurements as an internal lookup table!!

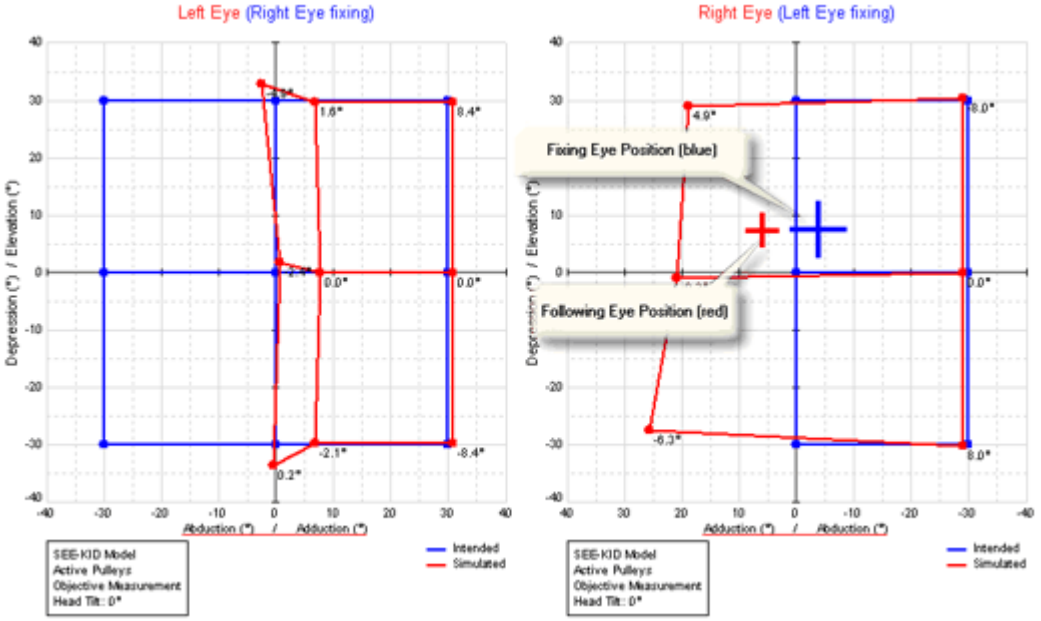
Instead, **SEE++ moves pulleys as a function of eye position and muscle force** and therefore **produces comparable results measured by Kono et al.** However, SEE++ currently does not differentiate between orbital and global layers, a task which is currently under development.

In May of 2005, Joel Miller provided the following list of suggestions under the title "fix obvious SEE++ problems". We have tried to incorporate these suggestions where it was reasonable to us. We would like to respond "graphically" where possible:

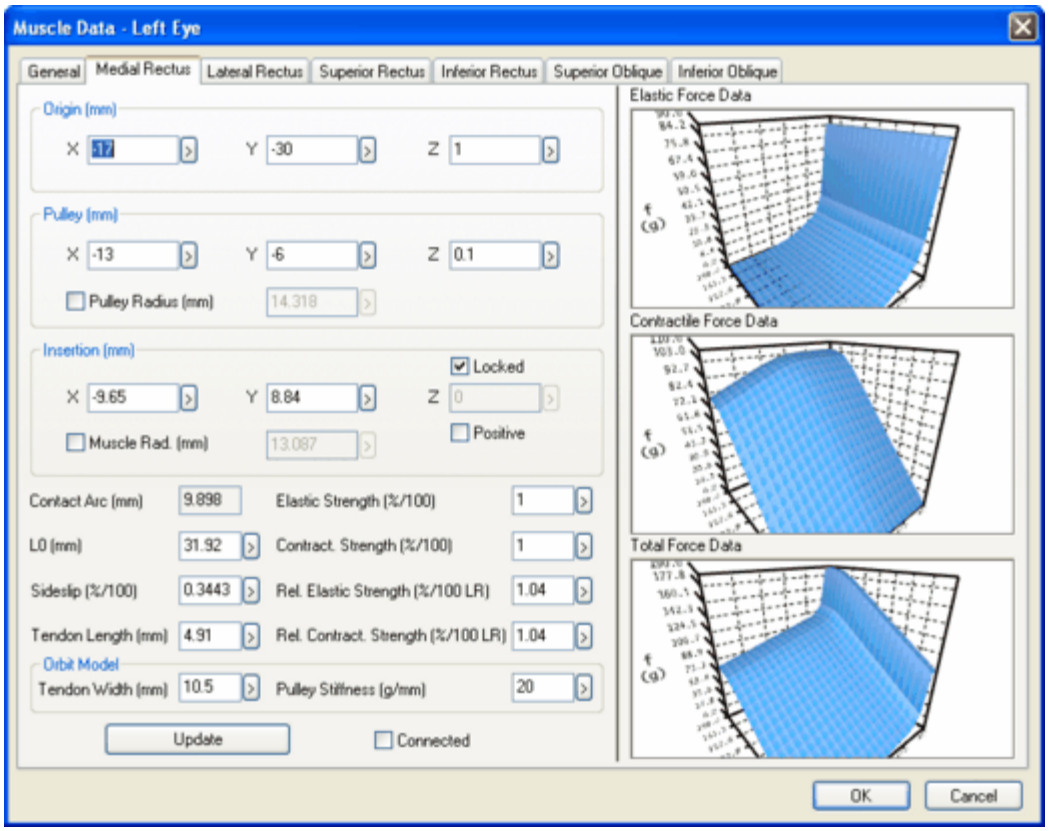
1. ***"3D Eye: "Coordinate Axes" are unspecified as to reference, though they are apparently head-fixed. They are unlabeled & unsigned."***



2. ***"3D Eye & Hess Chart: Fixing & Following positions of 3D Eye are shown on the Hess Chart (fixing eye on one chart and corresponding following eye on the other) inconsistently with how Fixing & Following positions are shown within a Hess Chart (corresponding fixing and following eyes on same chart)."***



3. ***"Muscle Data: Muscle Strength: I know what you mean here from Orbit, but I'll bet it's confusing to almost everyone who is not an Orbit user."***

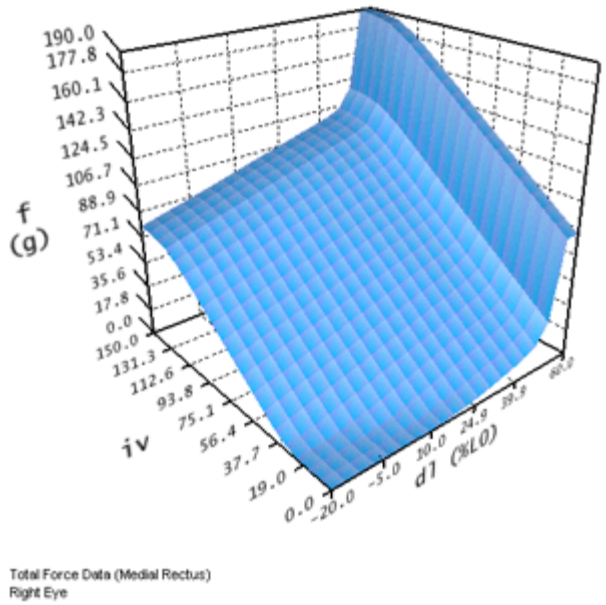


The „Muscle Strength“-parameter was eliminated with SEE++ Version 6!

4. ***"Muscle Data: Total Strength: This is a bad idea. It indulges users who can't distinguish elastic from contractile muscle force. This parameter also breaks the graphical connection between the 3 force surfaces."***

The „Total Strength“-parameter was eliminated with SEE++ Version 6!

5. ***"Muscle force surfaces have undefined & ilegible axes & units. Providing 3D rotation of these graphs is just eye candy, offering opportunities for confusion."***



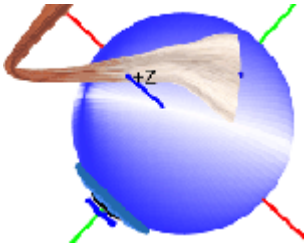
Total Force Data (Medial Rectus)
Right Eye

Muscle force surfaces have axes and units since SEE++ Version 6.1. 3D rotation and zoom was not omitted due to the fact that it can be useful when exporting images.

6. ***"Muscle Force Distribution: Why the bias in favour of showing variation with horizontal gaze? Shouldn't this be some sort of 3D plot that treats equally, at least horizontal and vertical gaze components?"***
   This diagram has been traditionally used in some publications and is partially familiar to clinicians. Showing 3D-plots would definitely lead to confusion and simultaneously render the diagram difficult to understand. The diagram shows horizontal gaze ranges in accordance with most textual explanations in literature where muscle pulling directions are listed with their changing functions when the eye is ab- or adducted.

7. ***"Hess Diagram: Why don't you show torsion graphically? Even old-fashioned Orbit does!"***
   Torsion is shown textually next to each simulated eye position in SEE++, a representation that we think is sufficient and easy to understand since Orbit™ needs to multiply torsion in order to show it graphically.

8. ***"Functional Topography" is undefined. I find no mention of it in the User's Manual. If it is actually something interesting & useful, it must be clearly explained.***
   Currently, the SEE++ user's manual is unfortunately not up to date and we are currently working on a new version. A transcript of the homepage text follows:



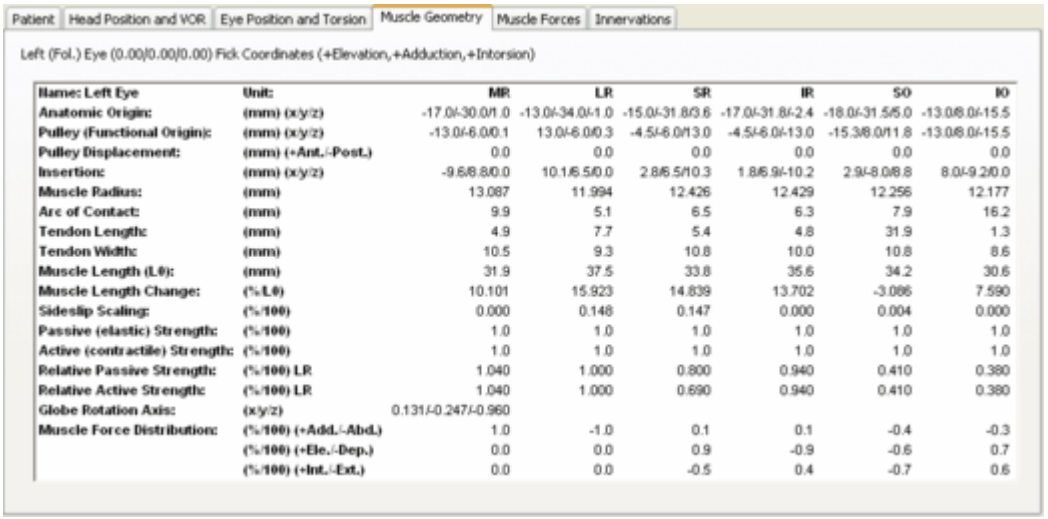
   The "functional topography" in SEE++ makes it possible to transform the muscle force distribution for a muscle and its possible insertions on the globe into a color coded 3D visualization. Therefore, the insertion of a selected muscle is "virtually" moved around the globe and at each location, the directional components (ab-/adduction, elevation/depression, in-/extorsion) are projected onto the surface as color-coded magnitude (darker colors indicate the

dominant direction of pull).

9. ***"Eliminate the "Pulley Model", which as I understand, is some sort of rudimentary Orbit-like model. It is nowhere described or validated, and causes only worry and confusion."***
   The „Pulley Model“ is now called **„SEE-KID Model“** and simulates **static non-moving pulleys** and forms the basis for the **“SEE-KID Active Pulley Model”** which **simulates the active (coordinated control) pulley hypothesis**.

10. ***"Where is the equivalent of Orbit's "Mechanical State Viewer", which showed simulation details? How are we to learn about the underlying mechanics without it? Its absence makes it impossible to really compare the SEE++ implementation of "Orbit core" with that of Orbit 1.8. Its absence causes worry and doubt! This level of insight into the model's functioning is essential!"***

We totally agree!


Patient | Head Position and VOR | Eye Position and Torsion | Muscle Geometry | Muscle Forces | Innervations

Left (Fol.) Eye (0.00/0.00/0.00) Fick Coordinates (+Elevation,+Adduction,+Intorsion)

Name: Left Eye	Unit:	MR	LR	SR	IR	SO	IO
Anatomic Origin:	(mm) (x/y/z)	-17.0/-30.0/1.0	-13.0/-34.0/-1.0	-15.0/-31.8/3.6	-17.0/-31.8/-2.4	-18.0/-31.5/5.0	-13.0/8.0/-15.5
Pulley (Functional Origin):	(mm) (x/y/z)	-13.0/-6.0/0.1	13.0/-6.0/0.3	-4.5/-6.0/13.0	-4.5/-6.0/-13.0	-15.3/8.0/11.8	-13.0/8.0/-15.5
Pulley Displacement:	(mm) (+Ant./-Post.)	0.0	0.0	0.0	0.0	0.0	0.0
Insertion:	(mm) (x/y/z)	-9.6/8.8/0.0	10.1/6.5/0.0	2.8/6.5/10.3	1.8/6.9/-10.2	2.9/-8.0/8.8	8.0/-9.2/0.0
Muscle Radius:	(mm)	13.087	11.994	12.426	12.429	12.256	12.177
Arc of Contact:	(mm)	9.9	5.1	6.5	6.3	7.9	16.2
Tendon Length:	(mm)	4.9	7.7	5.4	4.8	31.9	1.3
Tendon Width:	(mm)	10.5	9.3	10.8	10.0	10.8	8.6
Muscle Length (L0):	(mm)	31.9	37.5	33.8	35.6	34.2	30.6
Muscle Length Change:	(%L0)	10.101	15.923	14.839	13.702	-3.086	7.590
Sideslip Scaling:	(%/100)	0.000	0.148	0.147	0.000	0.004	0.000
Passive (elastic) Strength:	(%/100)	1.0	1.0	1.0	1.0	1.0	1.0
Active (contractile) Strength:	(%/100)	1.0	1.0	1.0	1.0	1.0	1.0
Relative Passive Strength:	(%/100) LR	1.040	1.000	0.800	0.940	0.410	0.380
Relative Active Strength:	(%/100) LR	1.040	1.000	0.690	0.940	0.410	0.380
Globe Rotation Axis:	(x/y/z)	0.131/-0.247/-0.960					
Muscle Force Distribution:	(%/100) (+Add./-Abd.)	1.0	-1.0	0.1	0.1	-0.4	-0.3
	(%/100) (+Ele./-Dep.)	0.0	0.0	0.9	-0.9	-0.6	0.7
	(%/100) (+Int./-Ext.)	0.0	0.0	-0.5	0.4	-0.7	0.6

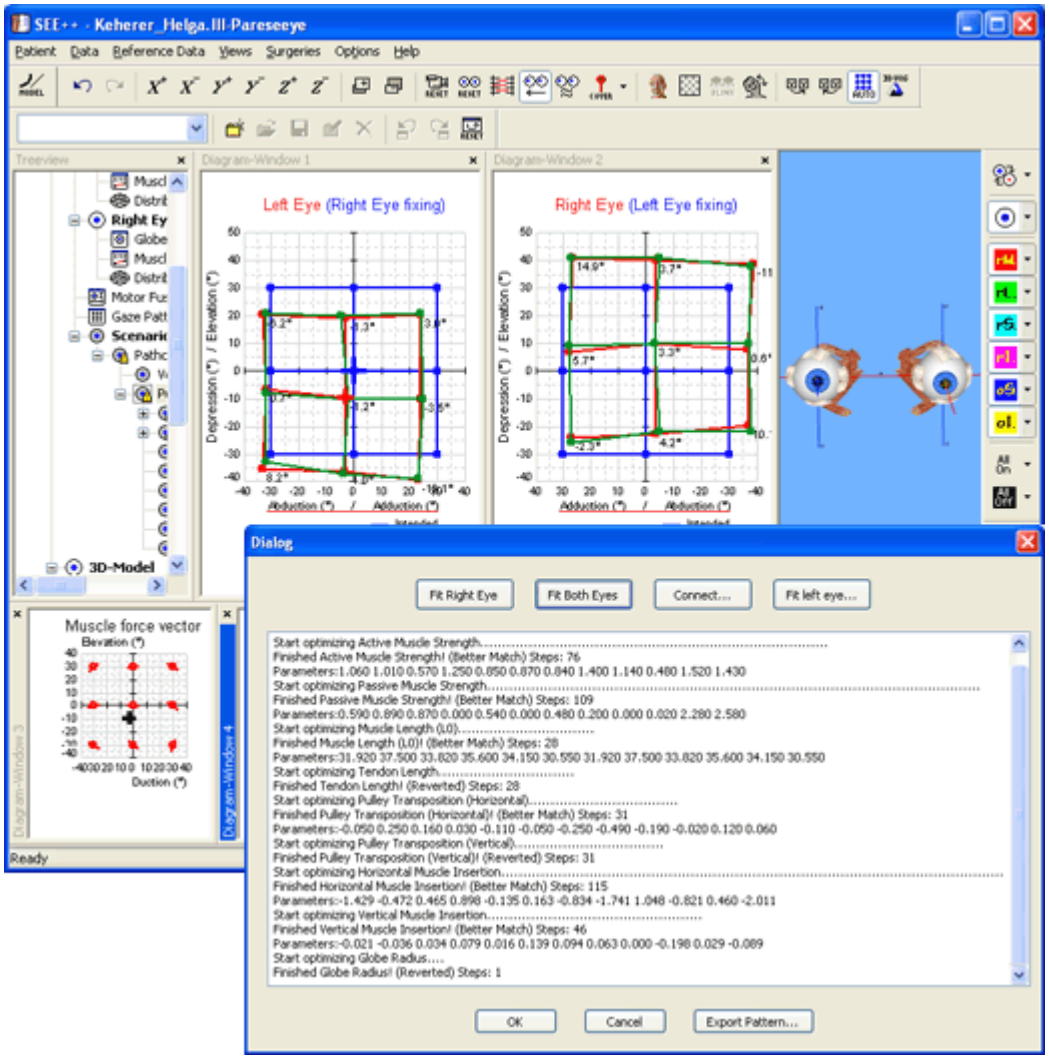
11. ***"The "Muscle Force Vector" diagram should probably be eliminated. If I correctly guess what it is, it is pretty meaningless if more than 1 muscle is selected. Where is this diagram explained?"***
    The „Muscle Force Vector” diagram shows rotational axes in different eye positions for selected muscles. The diagram still exists, although we agree that it should be replaced by somewhat more meaningful.

Miller made the following suggestions for improvements:

1. ***"User Levels in Orbit was a good idea! Why did you abandon this helpful interface concept? It makes it easy to include full details for advanced users (and for anyone who doubts the program's credibility!), while providing a simple interface for less knowledgeable users."***
   SEE++ is designed to be used in different levels of detail without omitting menu items or program functions. It is however a fact that users need to learn how to use the software, which from our point of view holds for almost any computer program. Furthermore, we think that using different modes for the user interface is the wrong approach from the point of usability. **The GUI design should be simplified but not specialized!** A well structured user interface does not

confuse expert users, while giving the novice an excellent chance to learn and understand how the program works.

2. ***"Where is the "Parameter Fitter"? That was a very cool feature of Orbit! Smart guys like you can certainly implement a fitting method better than Orbit's "brute force" search."***
   For the simulation of simple pathological cases, an automatic generator for parameter values is included in SEE++. Therefore, the angle of squint in primary position is entered and the system finds parameter values for simple muscle palsies, gaze palsies or concomitant forms of squint. Currently, there exists a prototype for the parameter-fitter for measured patient data and it will be integrated into SEE++ in the next phase of development.



We are smart guys, but we can't do magic!

2. ***"Orbit had extensive help, including "balloon help" (sort of like Window's little yellow "tool tips" tags), which tried to assure that the meaning of every item in the interface was clear. Is the meaning of everything in the SEE++ interface clear? Links to sections of the User Manual are not enough!"***
   Since SEE++ is Windows-Software, it uses Windows user interface guidelines which fortunately do not provide such overloaded, interfering and annoying things!

3. ***"Implement Orbit's "steepest descent" solution method as an alternative to SEE++'s Marquart-Levenberg method, to facilitate comparisons with Orbit 1.8."***
   This is impossible since SEE++ uses a non-linear optimization approach where all values are

absolutely calculated in each simulation step. In Orbit™, optimization proceeds iteratively in that each parameter is modified relative to the previous iteration step until a minimum is reached. To resolve any misunderstandings, the Orbit™ implementation in SEE++ uses exactly the same optimizer as the Orbit™ 1.8 Macintosh application, Levenberg-Marquardt optimizer is only used for the SEE-KID biomechanical models (SEE-KID Model and SEE-KID Active Pulley Model).

4. ***"You need to clarify the relationship of SEE++ to Orbit 1.8. We have spoken about this in detail."***
   Basically, we see SEE++ as replacement or extension to Orbit™ with more clinical relevance while providing essential functionality similar to what Orbit™ offers. **For detailed information we refer to our analysis of Orbit™.**

5. ***"The SEE++ software architecture utilizes the Orbit 1.8 core code in a somewhat different way than does Orbit 1.8 itself. There are reasons to worry that these differences limit SEE++ validity to relatively "normal" eyes! Explain differences clearly and completely. Characterize resulting differences in simulations, including the magnitudes & situations in which these differences occur. It's not fair to simply feature simulations in which these differences do not show up!"***
   This is an extended topic and we refer to our analysis of Orbit™.

Miller's Mid-Term Suggestions: Target Specific Problems in Strabismus and EOM Coordination, such as

1. ***"So-called muscle "overaction" syndromes. For each, you will need to get several sets of patient data with good alignment measurements, preferably before & after well-described surgery. You can probably show that such syndromes are mainly or totally due to abnormalities of elastic force, not contractile force."***
   This is a nice idea and we are going for that in the near future. We have already published a paper on the simulation of the "**Heavy Eye**" Syndrome (**Model Based Improvements in the Treatment of Patients with Strabismus and Axial High Myopia, IOVS;48:1133-1138**).

Miller's Long-Term Suggestions: Leverage SEE++'s Modern Software Architecture

1. ***"Produce a new, conceptually unified, User Interface, not bound to Windows graphics (probably in Java)"***
   This can be done by anyone and we have shown an example of a **Java-Applet on our homepage**. The **SEE++ Calculation Server** architecture provides open flexibility for almost any platform, and the **SEE++ MatLab Interface** offers even more possibilities to experiment.

2. ***"Use the SEE++ container architecture to provide access to truly new biomechanical models"***
   We certainly consider our SEE-KID biomechanical models „truly" new, however, for the development of new models there is the **SEE++ Calculation Server** architecture and the **SEE++ MatLab** Interface available.

3. ***"Active pulley models"***
   We have implemented a first version of the **coordinated control hypothesis of Active Pulleys** and will continue working on that topic.

4. ***"Calculate, rather than assume, Sherrington's Law of Reciprocal Innervation"***
   We do it the **same way as Orbit™ at the moment**, but we have included this in our ToDo-List.

5. ***"Include structures, such as the "neurofibrovascular bundle" (Stager, 1996), the***

lateral levator aponeurosis and other prominent intermuscular tissues"
We have included this in our ToDo-List.

6. ***"Explicitly include target distance, vergence input. Is L2 an emergent property?"***
   We have included this in our ToDo-List.

7. ***"Include otolith input"***
   We experimented with **simulating VOR and head-tilt** and want to continue to provide **more sophisticated models of brainstem and VOR**.

8. ***"Explicit implementation of common coordinate systems"***
   If we interpret this suggestion correctly, then the answer would be that SEE++ programmatically offers the possibility to change coordinate systems.

9. ***"Allow non-spherical (particularly deeply myopic) eyes"***
   For the simulation aspect, this has already been achieved as shown in Model Based Improvements in the Treatment of Patients with Strabismus and Axial High Myopia (**IOVS;48:1133-1138**). We have tried to **visualize non-spherical globes in 3D**, but this is currently in a beta stage, further development on hold.

10. ***"Implement common muscle-splitting operations"***
    A **major goal of the SEE-KID** project which we plan to finish in 2008.

11. ***"Make muscle surfaces part of the biomechanical model (not just for show)"***
    This is definitely a **prerequisite to simulating splitting surgeries** and global and orbital layers of muscles.

Last Updated (Thursday, 07 June 2007)