Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
225157	0	0	0														
270034	10.5	10	0.5	1	3	3	3		One test commit doesn't count as using version control. Rot and scale: Scale applied to diagonal of identity matrix. Create 3 matrices R, S, T, get compound xform modelToWorld=T*R*S	0	0.5						
293846	11.5	10	1.5	1	3	3	3			1						0.5	Animation (0.5)
295323	0	0	0														
345642	0	0	0														
348843	0	0	0														
349936	16	10	6	1	3	3	3		VCS: please provide a logfile next time for points. Perspective: we were looking to have the fov changes matched in both directions. Working version of the camera is very simple, cool physics though!			1		1	1.5	2.5	Animation (0.5p), Basic ASCII PLY loader (2.0p)
350475	11	10	1	1	3	3	3			1							
352091	0	0	0														
353980	9	7	2	1	1	3	2		R2: result of cross product not saved, tip not at origin. R4: f[4] and f[5] use same index. Also, going via string is unnecessary. Rotate works, but saving angles and recreating matrix every frame is preferred over accumulation	1	1						
354439	10	10	0	1	3	3	3		Git log missing, please submit with next assignment	0							
355593	0	0	0														
356026	13.5	9	4.5	1	2	3	3		R2: normals flipped. Viewport: not reset after glDrawArrays (breaks UI). Transformation order unintuitive: rotation and scaling don't happen in object space. Note: normals have w=0 in homogeneous coords (shader). Normals not normalized after transformation	1	1			1.5	0.5	0.5	Animation (0.5)
361749	0	0	0														
369181	16	10	6	1	3	3	3		Missing git logfile, please submit with next assignment. Normals not normalized after transformation	0	1	0.5			2	2.5	Animation (0.5), PLY reader (2.0)
372660	3	3	0	1	2				R2 normals wrong. Extra adds nothing over R1								Transl. with mouse wheel
387370	0	0	0														
425575	6	6	0			3	3		Single commit doesn't count as version control	0							
425614	12	10	2	1	3	3	3			1	1						
426419	16.5	10	6.5	1	3	3	3		Viewport not reset after glDrawArrays (breaks UI), image not centered	1	1		2	2	0	0.5	Animation (0.5)
427489	11	10	1	1	3	3	3			1							
428022	9	9	0	1	3	3	2		R2: please use provided constants (e.g. radius), positions assigned tuples of vec3s (first value ignored). R4: one-based indexing not corrected to zero-based								
429487	0	0	0														
430829	10	10	0	1	3	3	3		VCS proof not sufficient, please resubmit next round	0							
457598	12	10	2	1	3	3	3			1	1						

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
460297	0	0	0														
464772	0	0	0														
46477D	0	0	0														
46596K	0	0	0														
474199	15.5	10	5.5	1	3	3	3		Camera too simple for points (only new feature is distance control)		1		0	2	2	0.5	Animation (0.5)
474322	10	10	0	1	3	3	3		R1: you're right, would be better to recreate modelToWorld each frame from position (vec3) instead of accumulating								
474458	11	10	1	1	3	3	3			1							
474898	10	10	0	1	3	3	3										
475389	0	0	0														
475813	0	0	0														
475910	10	10	0	1	3	3	3		Git log missing, please submit with next assignment	0							
477329	10	9	1	1	2	3	3		R2: Cone normals inverted	1							
477811	11	9	2	1	2	3	3		R2: normals too complicated (and incorrect), just use cross product of two tri edges	1	1						
478328	12	10	2	1	3	3	3			1	1						
478470	0	0	0														
478687	0	0	0														
479505	12	10	2.5	1	3	3	3	-0.5	VS project files missing (-0.5)	1	1					0.5	Animation (0.5)
479589	11	10	1	1	3	3	3		Single git commit doesn't count as version control	0	1						
479741	17	10	7	1	3	3	3		Scaling in object space (T*R*S) would make more sense than in world space (T*S*R). Simple camera. Normals not normalized after transformation	1	1		1	1.5	2	0.5	Animation (0.5)
480086	0	0	0														
480248	0	0	0														
480714	12	10	2	1	3	3	3		Scaling along world x makes less sense than obj. x. Normal xform: inverse transpose of matrix not used	1	1	0					
480798	10	10	0	1	3	3	3		Normal xform not using inverse tranpose, normal also not normalized after transform			0					
481577	12	10	2	1	3	3	3			1	1						
493840	15	9	6	1	3	3	2		R4: obj loader fails for garg. Single commit doesn't count as version control. Simple camera. Viewport: should reset after glDrawArrays to fix UI	0	1	1	1		0.5	2.5	Animation (0.5), stl loader (2.0)
506300	13.5	10	3.5	1	3	3	3		Simple camera. Note: undo viewport change after glDrawArrays to fix UI.	1	1		1		0.5		
508285	12	10	2	1	3	3	3			1	1						
51620U	0	0	0														
524926	11	10	1	1	3	3	3			1							

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
525417	12.5	9	3.5	1	2	3	3		R2: normals inverted. Normals not normalized after transofmation. Rotation: you can use Mat3f::rotation() instead of writing formulas by hand. Disappearing objects caused by the so called clipping planes (fnear, ffar). Camera: only distance not enough for points. Please make your repositories private on GitHub!	1	1		0	1.5			
525491	7	7	0	1	3	3											
525941	0	0	0														
526050	13	10	3	1	3	3	3		Undo viewport change after glDrawArrays to fix UI	1	1				0.5	0.5	Animation (0.5)
526319	10	10	0	1	3	3	3										
526775	1	1	0	1													
527143	0	0	0														
527389	0	0	0														
528867	0	0	0														
528883	9.5	10	0	1	3	3	3	-0.5	VS project files missing (-0.5)								
529196	0	0	0														
529303	5	5	0	1	2	2			R2: cone tip not in origin. Cross product broken, use tri edges (p2-p0) and (p1-p0). R3: normals read from positions vector.								
530185	0	0	0														
530363	0	0	0														
530619	0	0	0														
530648	0	0	0														
530868	13.5	10	3.5	1	3	3	3		Normal xform is correct. Simple camera. Normals not normalized after transformation	1	1	0.5	1				
530981	11	9	2	1	2	3	3		R2: cone position and orientation wrong.	1	1						
540094	0	0	0														
540654	0	0	0														
544566	9	7	2	1	0	3	3		R2: tip not at origin, coords wrong (you need positions at i and i+1), normals missing. Scale and rot: rot. axis wrong, scale is uniform (not non-uniform along x). Animation logic seems wonky, sometimes stops on its own	1	0.5					0.5	Animation (0.5)
549040	9	9	0	1	2	3	3		R1: OK, but recreating matrix each frame recommended over accumulating. R2: cone normals incorrect, use e.g. cross(v2-v1, v2-v0).norm()								
549163	9.5	10	0	1	3	3	3	-0.5	Single init commit not enough for VCS. Submission missing files, not self-contained (-0.5)	0							
55055P	20.5	10	10.5	1	3	3	3		Camera works well, but is not quite a trackball.	1	1		2	2	2	2.5	Animation (0.5p), Basic ASCII PLY loader (2p)
552794	11	10	1	1	3	3	3			1							
552969	0	0	0														

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
554598	20	10	10	1	3	3	3		Camera: not really traditional trackball. Also spazzes out when camera fwd aligns with y axis. Use window_.isKeyDown() in App::handleEvent for smoother movement. Persp: aspect ratio fix typically applied to x, not y. Normals not normalized after transformations	1	1		2	1.5	2	2.5	Animation (0.5), ASCII PLY (2.0)
563068	11	9	2	1	2	3	3		R2: normals inverted	1	1						
576149	10.5	10	1	1	3	3	3	-0.5	VS project files missing (-0.5)	1							
585716	0	0	0														
586333	12.5	10	2.5	1	3	3	3		You can initialize current_scale_ to 1 in the constructor of App, either in the initializer list, or in the body of the constructor. current_angle_ must also be initialized. Transformed normal not normalized	1	1	0.5					
586702	12	10	2	1	3	3	3			1	1						
586980	11.5	9	2.5	1	2	3	3		R2: normals inverted	1	1					0.5	Animation (0.5)
587316	0	0	0														
587471	13.5	10	3.5	1	3	3	3		Transformed normal not normalized. Simple camera.	1	1	0.5	1				
588289	16	10	6	1	3	3	3		Normals not normalized after transform. Simple camera.	1	1	0.5	1		2	0.5	Animation (0.5)
589291	0	0	0														
589343	0	0	0														
589848	12.5	10	2.5	1	3	3	3		Very simple camera. Scaling along global x perhaps less intuitive than object space x	1	1		0.5				
590921	0	0	0														
591904	10	10	0	1	3	3	3										
591946	13	10	3	1	3	3	3		No VCS log submitted. Very simple camera. Note: normal has w coordinate 0 in homog. coords (shader).	0	1	1	0.5			0.5	Animation (0.5)
592929	10	9	1	1	2	3	3		R2: cone normals inverted. Obj. rot: left-multiplying matrix causes rotation in world space around origin (not wrong, but less intuitive than obj. space rot). Also, recreating matrix from angles each frame is preferred over accumulating		0.5					0.5	Animation (0.5)
593274	10	10	0	1	3	3	3		Single commit not enought for VCS point.	0							
593847	0	0	0														
594435	0	0	0														
595926	10.5	9.5	1	1	2.5	3	3		R2: inverted normals.	1							
595997	0	0	0														
596747	0	0	0														
597429	10	10	0	1	3	3	3		R2: cone tip not in origin (-0p)								
597623	12.5	9.5	3	1	2.5	3	3		R2: Inverted normals.	1	1		0.5			0.5	Animation (0.5p)
597937	20	9	11	1	2	3	3		R2: normals inverted. Simple camera. Aspect ratio fix typically applied to x axis (y is not incorrect though). Simplification: boundaries not preserved, in addition to problems mentioned in readme. Still works reasonably well	1	1		1	2	2	4	Animation (0.5), Simplification (3.5)
602851	0	0	0														

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
602893	13	10	3	1	3	3	3		Single git commit not proof enough, please resubmit next round. Scaling in object space (before rotation) would make more sense than scaling along the global x axis. Simple camera. Undoing viewport changes after glDrawArrays would fix UI	0	1		1		0.5	0.5	Animation (0.5)
603096	11	10	1	1	3	3	3			1							
603245	11	10	1.5	1	3	3	3	-0.5	VS project files missing. Simplification: some points given for good start							1.5	Simplification attempt (1.5p)
604095	24	10	14	1	3	3	3		VCS: one commit doesn't really count. Camera not quite a virtual trackball. Simplifier: please always provide a reasonable way of controlling parameters like the simplification fraction here. Simplifier is slow but yields okayish results. Some interesting variable names?	0	1		2	2	2	7	Animation (0.5p) Simple ASCII PLY loader (2.0p). Simplifier (4.5p)
604273	11.5	9.5	2	1	2.5	3	3		R2: inverted normals. VCS: a couple of commits is hardly version control. Rotate&Scale: you don't seem to have included the changing parameters to the modelToWorld matrix. In similar fashion to the translation params. The .vs folder is not needed in the submission, so no worries!	1					0.5	0.5	Animation (0.5p)
606064	10	10	0	1	3	3	3		Single git commit not enough for VCS point, please resubmit longer log next round	0							
608949	0	0	0														
609155	0	0	0														
609249	0	0	0														
612472	12.5	10	2.5	1	3	3	3			1	1					0.5	Animation (0.5p)
612498	0	0	0														
612870	0	0	0														
614577	0	0	0														
614580	0	0	0														
621308	9.5	8	1.5	0.5	3	3	1.5		R1: don't include worldToClip in your modelToWorld matrix. This fixes the flattened look of all models. R4: face data filled in wrong order, you want to fill it sequentially from iss (with sinks in between), i.e. positions and normals are interleaved.	1	0		0.5				
628835	11	10	1	1	3	3	3			1							
63036R	0	0	0														
641922	0	0	0														
646804	19.5	9.5	10	1	2.5	3	3		R2: normals inverted. Normal transformation: quite sneaky to multiply the normal vector from the right to avoid taking a transpose. Camera is not quite a trackball. Perspective lacking a fov-slider.	1	1		2	2	1	3	Animation (0.5p) PLY loader with binary support (2.5p)
647764	11	10	1	1	3	3	3			1							
648530	0	0	0														
648860	0	0	0														
650191	0	0	0														

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
650227	9	9	0	1	2	3	3		R2: Incorrect normals. Tip not at origin: can't use positions directly in cross-product since the third vertex is not at the origin.								
650405	11	10	1	1	3	3	3			1							
650560	20.5	10	10.5	1	3	3	3		PLY: it makes most sense to first read the header metadata into custom structs, and then process the vertex and normal data. Simpl: still quite far from a working solution	1	1		3	2	2	1.5	Animation (0.5p), Simplification (1.0)
650942	14	9	5	1	2	3	3		R2: you need to use the cross product to get the normals. It is a bit unfortunate that the framework allows to init a Vec3 with the scalar you get from the dot-product. VCS: could not find the logfile you mentioned in the readme, return with next assignment. Rotation causes the object to move. Camera is quite simple.		0.5		2		2	0.5	Animation (0.5p)
651527	11	10	1	1	3	3	3		VCS: A couple of commits within ~10mins isn't really version control...	1							
651585	8.5	8	0.5	1	3	1.5	2.5		R3: normals not handled. R4: normal indices also have to be shifted by -1. Change shading mode with T to test these							0.5	Animation (0.5p)
651637	10	9	1	1	2	3	3		R2: Inverted and unnormalized normals.	1							
651789	0	0	0														
652102	12.5	10	2.5	1	3	3	3		Normals transformed with the vanilla modelToWorld matrix, instead of the inverse-transpose.	1	1	0			0.5		
652131	9.5	9.5	0	1	2.5	3	3		R2: just inverted normals - that's why the lighting looks different!								
652335	17.5	10	7.5	1	3	3	3		Normals not normalized after transform. Camera quite simple, not quite trackball. Perspective: rescaling window works, but fov changing is somewhat broken.	1	1	0.5	1.5		1	2.5	ASCII PLY loader with fan triangulation.
652649	12	10	2	1	3	3	3									2	
652898	0	0	0														
652937	10	10	0	1	3	3	3										
653127	13.5	10	3.5	1	3	3	3		Normals not normalized after transformation	1	1			1.5			
653596	0	0	0														
653693	10	10	0	1	3	3	3										
653871	0	0	0														
653907	10	10	0	1	3	3	3										
653910	10	10	0	1	3	3	3										
654595	20.5	10	10.5	1	3	3	3			1	0.5		3	2	2	2	Simple binary STL (1.5), Animation (0.5)
655057	11	10	1	1	3	3	3		VCS: a couple of commits is hardly version control.	1							
655086	27.5	10	17.5	1	3	3	3		Neat camera! We try to read the readmes, yes: no immediate tricks or tips to improve, but your implementations seems to eat unreasonable amounts of memory. Maybe try to cut that down? Parallelism might also help. Results look quite nice on the ball and the torus.		1		3	2	2	9.5	Animation (0.5p) Simple ASCII PLY loader (2.0p) Simplifier (4.0p). Rotations with quaternions (3.0p)
655109	10	10	0	1	3	3	3		VCS: a single commit does not really count.								

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
655251	11.5	10	1.5	1	3	3	3		Trackball attempt.				1.5				
655264	5	5	0	1	2	2	0		R2: you need to use the cross-product to find the normals. This is different from element-wise multiplication. R3: you are discarding the face-information completely. This hard-coded version works only for the test-tetrahedon and does not get the normals right.								
655471	11.5	10	1.5	1	3	3	3		mat3[0] accesses the first column of the matrix, so your mat--vec product is actually transpose(mat)--vec product. Hence the odd behavior. The result is also unnormalized. You don't actually need to do this elementwise by hand, rather use the built-in operators!		1	0.5					
655691	0	0	0														
655853	11.5	9.5	2	1	2.5	3	3		R2: Normals inverted.	1	1						
656250	9	9.5	0	1	2.5	3	3	-0.5	VS project files missing (-0.5). R2: triangles intersecting; you can use 'middle' shifted down as v1.pos directly								
656454	17	10	7	1	3	3	3			1	1		2	2	0.5	0.5	Animation (0.5p)
656616	9.5	3.5	6	1	2.5				R2: inverted normals. VCS: a single commit does not really count. Normal transform: not normalized after transform.	0	1		2	1.5		1.5	STL-loader (1.5p)
657291	14	10	4	1	3	3	3			1	1			2			
657314	0	0	0														
657327	10	10	0	1	3	3	3		VCS: a couple of commits within few minutes does not really count.	0							
657482	0	0	0														
657796	11.5	10	1.5	1	3	3	3		Simple camera.				1	0.5			
657893	12	10	2	1	3	3	3			1	1						
659914	14.5	10	4.5	1	3	3	3			1	1			2	0.5		
660246	19.5	10	9.5	1	3	3	3		Simple camera	1	1		1	2	2	2.5	PLY (2p), Animation (0.5p)
660877	0	0	0														
660893	0	0	0														
663191	0	0	0														
663272	0	0	0														
665380	14.5	10	4.5	1	3	3	3		VCS: please add the logfile to the submission next round to get points. Normal transformation: uModelToWorld used directly, instead of its inverse-transpose.		1	0	2		1	0.5	Animation (0.5p)
665898	11	10	1	1	3	3	3			1							
666172	12.5	10	2.5	1	3	3	3		Your rotation is for the camera, not the object. Scaling causes the object to move. Simple camera.	1	0		1			0.5	Animation (0.5p)
666350	15.5	10	5.5	1	3	3	3		Normal transform: unnormalized after transform.	1	1	0.5	3				
666680	13	10	3	1	3	3	3			1	1	1					
667249	12	9.5	2.5	1	2.5	3	3		R2: inverted normals. VCS: could not find the logfile in the submission, submit with next assignment. Normal transform: you are only taking the inverse, not transposing.		1			1.5			

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
67137M	3	3	0	1	2				R2: normals inverted. Note: you can create vectors in one go: U = v1 - v0, no need to create elementwise								
67627H	12	10	2	1	3	3	3			1	1						
677734	0	0	0														
678089	0	0	0														
68933B	0	0	0														
69247N	0	0	0														
700436	0	0	0														
705570	26.5	10	16.5	1	3	3	3		Simplifier robust and not unreasonably inefficient.	1	1		3	2	2	7.5	Animation (0.5p), STL-loader (2.0p), Simplifier (5p)
706566	9.5	9.5	0	1	2.5	3	3		R2: inverted normals. Please comment out debug-related std::couts for the submission. Mesh loading was slow because of these.								
708784	18	10	8	1	3	3	3		Normal transform: normal unnormalized after transform. Camera works well, but is not quite a virtual trackball. We can give you points for the build-system if you are prepared to share and document your approach with a future assignment.	1	1		2	1.5	2	0.5	Animation (0.5p)
708904	0	0	0														
708920	11	10	1	1	3	3	3			1							
708933	0	0	0														
709291	12	10	2	1	3	3	3		R1: motion only in 1D (-0p). Simplifier flips normals, tends to only give spherical outputs and hardly works at all for more complicated models.							2	Simplifier attempt (2p)
709628	11.5	10	1.5	1	3	3	3		VCS: a single commit is not really version control.		1					0.5	Animation (0.5p)
710086	12	10	2	1	3	3	3		VCS: please include a logfile in the submission for the point!		1	1					
710497	18.5	10	8.5	1	3	3	3			1	1		2	2	2	0.5	Animation (0.5p)
710743	13.5	10	3.5	1	3	3	3		R1: translation only in 2D (-0p). R2: Cone has only half the expected faces (-0p). Normal not normalized after transform. Simple camera.	1	1	0.5	1				
710976	11	9.5	1.5	1	2.5	3	3		R2: inverted normals. VCS: could't find the logfile mentioned in readme. Submit with next assignment, please!		1					0.5	Animation (0.5p)
711182	11	10	1	1	3	3	3			1							
711467	18.5	10	8.5	1	3	3	3		Normal transform: normal not normalized after transform.	1	1	0.5	3		0.5	2.5	Animation (0.5p), Basic ASCII PLY loader (2.0p)
711551	17.5	10	7.5	1	3	3	3		Camera is quite simple. Normal transform matrix is missing the transpose. Viewport is still more-or-less the simpler, alternative - you are not touching the perspective matrix.	1	1		1	1.5	0.5	2.5	Animation (0.5p), Basic ASCII PLY loader (2.0p)
711810	3	3	0	1	2	0	0		R2: you are computing the cross product using v1.normal-v0.position, which at this point is always the zero vector. As a result, all the normals are zero vectors.								
711904	0	0	0														
712550	10	10	0	1	3	3	3										

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
712686	0	0	0														
712819	9	9	0	1	2	3	3		R2: You are trying to compute the normals before having computed the vertex positions - using zero-initialized vectors, resulting in all zero normals.								
712958	0	0	0														
713672	0	0	0														
714985	12	9.5	2.5	1	3	2.5	3		R3: You are reading the normals from the positions-vector instead of the normals-vector. Indexing is correct, though. Normals not normalized after transform.		1			1.5			
716080	10	10	0	1	3	3	3										
716462	13.5	10	3.5	1	3	3	3		Normal not normalized after transformation. Camera is quite simple.	1	1	0.5	1				
716718	0	0	0														
716860	13	10	3	1	3	3	3		Normal not normalized after transform.	1	1	0.5				0.5	Animation (0.5p)
717377	13	10	3	1	3	3	3		Simple camera.	1	1		1				
717474	0	0	0														
717513	12.5	10	2.5	1	3	3	3		R1: movement only in 2D (-0p). Normals not normalized after transformation.	1	1	0.5					
717539	11.5	10	1.5	1	3	3	3		Wrong order in the composition of the modelToWorld matrix.	1	0.5						
718020	0	0	0														
718871	9	9	0	1	2	3	3		R1: movement only in 2D (-0p). R2: incorrect normals								
722427	0	0	0														
723691	10	10	0	1	3	3	3		VCS: please submit a logfile next time for points.								
723905	0	0	0														
728667	17.5	10	7.5	1	3	3	3		Trackball: getting some unexpected behavior with some angles. Code looks reasonable. Perspective: no fov-slider. Normal transforms: normals not normalized after transform.	1	1		2.5	1.5	1	0.5	Animation (0.5p)
728900	10	10	0	1	3	3	3		VCS: a single commit isn't really version control...	0							
729132	9.5	10	0	1	3	3	3	-0.5	Next time, please also include the VS-project files in the submission! R1: translation only in 2D (-0p). A single commit isn't really version control... Could not find code for the extras you described in the readme.								
729967	10	10	0	1	3	3	3										
730309	0	0	0														
732080	0	0	0														
732255	0	0	0														
732323	0	0	0														
732336	0	0	0														
732352	0	0	0														
732459	14	10	4	1	3	3	3		Normal transform: you are using the vanilla modelToWorld matrix instead of the inverse-transpose. Camera works, but is very simple.	1	1	0	1		0.5	0.5	Animation (0.5p)

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
76509T	22.5	10	12.5	1	3	3	3		Simpl: Super fast! But boundaries not preserved. Normals not normalized after transformation		1			1.5	2	8	Simpl. w/ acc. structure (8p)
765510	16	9.5	6.5	1	2.5	3	3		R2: inverted normals. Camera works well, but is quite simple.	1	1		1.5	2	0.5	0.5	Animation (0.5p)
766331	0	0	0														
767042	0	0	0														
767136	12	10	2	1	3	3	3			1	1						
768504	10	10	0	1	3	3	3		R1: translation in 2D (-0p)								
769396	14	10	4	1	3	3	3			1	1	1			0.5	0.5	Animation (0.5p)
77388B	9.5	8.5	1	1	1.5	3	3		Cone: normals inverted, +1 in wrong place inside cos and sin	1							
779124	0	0	0														
780058	11	10	1	1	3	3	3			1							
780346	10	10	0	1	3	3	3										
782917	0	0	0														
783563	0	0	0														
783709	0	0	0														
786667	0	0	0														
78708M	0	0	0														
787543	0	0	0														
787640	0	0	0														
788380	0	0	0														
788678	0	0	0														
791982	0	0	0														
795700	0	0	0														
795755	13.5	9.5	4	1	2.5	3	3		R2: inverted normals. Normals unnormalized after transform, hence the bug you described.	1	1	0.5			1	0.5	Animation (0.5p)
796039	0	0	0														
804183	19.5	10	9.5	1	3	3	3		Readme is way more detailed than what is expected. The details are appreciated though!	1	1		3	2	2	0.5	Animation (0.5)
829155	0	0	0														
838191	2	1	1	1	0	0	0			1							
83873J	0	0	0														
84308F	6	7	0	1	3	3	0	-1	No readme, no VS project files (-1p). R2: normal formula incorrect (you want diff of positions, not sum), but happens to be correct here (-0p). R4: x,y,z coords of positions and normals treated as separate vectors. Face logic too complicated, just read into f from iss.								
84858E	0	0	0														
848754	15	10	5	1	3	3	3		Normal transforms: you are using the vanilla modelToWorld instead of the inverse-transpose. Result is also unnormalized. The camera is not quite a virtual trackball, unexpected results at some viewing angles.	1	1	0	2		0.5	0.5	Animation (0.5p)

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
875170	16.5	10	6.5	1	3	3	3		Normal transform: normal not normalized afterwards. Simple camera. Perspective: No fov-slider. PLY-loader: the normals you compute from the vertices are not coherent and flip within a single face.	1	1	0.5	1		1	2	Animation (0.5p). Very simple PLY loader (1.5p)
875251	11	9.5	1.5	1	2.5	3	3		R2: inverted normals.		1					0.5	Animation (0.5p)
875303	0	0	0														
875617	0	0	0														
876399	0	0	0														
877107	0	0	0														
877152	12	9.5	2.5	1	2.5	3	3		R2: normals inverted. Normal transform: in my editor your code had special characters "transpose(inverse£¨uModelToWorld£©)" in the shader which caused the crash. I wonder if these were not visible to you? Camera works somehow, but is very, very clunky. Some of the camera rotation parameters are seemingly uninitialized and resulted in really weird initial views sometimes.	1	1	0	0.5				
878591	10.5	9.5	1	1	2.5	3	3		R1: translation only in 2D (-0p). R2: inverted normals.	1							
878627	12	10	2	1	3	3	3		Normal transform: normal not normalized afterwards. Rotate&Scale: order of operations weird: object moves with scaling.	1	0.5	0.5					
878889	13	10	3	1	3	3	3		Normals unnormalized after transform. Viewport is completely broken due to not casting the integer window size to floats before division.	1	1	0.5			0	0.5	Animation (0.5p)
879105	14.5	10	4.5	1	3	3	3		Rotate&Scale: somewhat weird order of transformations, scaling moves the object. Good attempt on the trackball, however the camera is more-or-less unusable due to the flickering. Could not immediately see the problem.	1	1	1	1.5				
882134	0	0	0														
885128	9.5	9.5	0	1	2.5	3	3		R2: inverted normals.								
886648	16.5	10	6.5	1	3	3	3		Normal transforms: unnormalized normals after transform. Simple camera.	1	1	0.5	1		0.5	2.5	Animation (0.5p). STL-loader (2.0p)
889645	0	0	0														
892292	0	0	0														
898351	10.5	9.5	1	1	2.5	3	3		R2: inverted normals.	1							
899130	14	10	4	1	3	3	3		R1: movement only in 1D (-0p). Rotate&Scale: swap the order of rotate and scale to get rid of the weird effect. Camera not really a virtual trackball.	1	1		2				
900016	18.5	10	8.5	1	3	3	3		Normal transformation: normals unnormalized afterwards. Camera not quite a virtual trackball, some unexpected behavior with some viewing angles.		1		2	1.5	2	2	Animation (0.5p). Very simple ASCII PLY loader (1.5p)
901170	0	0	0														
901196	0	0	0														
913249	0	0	0														
913333	0	0	0														

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
913346	0	0	0														
913566	13.5	10	3.5	1	3	3	3		R1: translation in 1D (-0p). Normal transform: normal not normalized afterwards.	1	1	0.5			0.5	0.5	Animation (0.5p)
915221	7.5	8	0	1	3	3	1	-0.5	Next time, please also return the VS-project files (.sln etc)! There is a large-ish hidden .vs folder which you don't need to return, that most likely ate all the submission space. R1: translation only in 1D (-0p). R4: You are reading normals as positions and positions as normals + reading wrong values into the idx and sink variables. VCS: for points, please return a logfile in the next assignment.								
915250	9.5	10	0	1	3	3	3	-0.5	Next time, please also return the VS-project files! R1: translation only in 2D (-0p).								
917863	25	10	15	1	3	3	3		R1: translation only in 1D (-0p). Camera quite simple, not quite a trackball! Simplifier yields quite robust results but could still be a bit faster.	1	1		1.5	2	2	7.5	Animation (0.5p). Simple ASCII PLY loader (2.0p). Simplifier (5.0p)
918150	0	0	0														
918228	15	10	5	1	3	3	3		Please ensure that the button-instructions of the readme match those of your code! Rotate&Scale: you apply the scale to m00 and immediately divide it out, resulting in an identity transformation. Normal transform: normal not normalized afterwards. The way you compute the inverse-transpose of modelToWorld is quite involved and error-prone - just use mat3.inverse() etc. The math seems to check out, though!	1	0.5	1	1.5		0.5	0.5	Animation (0.5p)
918257	0	0	0														
918309	17.5	10	7.5	1	3	3	3		Your ZIP was actually a RAR! Normals not normalized after transformation	1	1		3	1.5	0.5	0.5	Animation (0.5p)
918396	9	9.5	0	1	2.5	3	3	-0.5	Please return a _filled_ readme! The TAs will not scour your code to figure out what is done. R2: inverted normals. VCS: a single commit is not really version control.	0							
918464	25.5	10	15.5	1	3	3	3		Perspective: matching in the sense that both horizontal and vertical fov be the same. Results in a kind of a zoom effect, rather than stretching. You were almost there! Simplifier: very slow for larger models and not that robust, good work on the boudary conservation!	1	1		3	2	1.5	7	Animation (0.5p) ASCII PLY loader with fan-triangulation (2.5 p). Simplifier (4.0 p)
918671	16	10	6	1	3	3	3		R1: movement only in 2D (-0p)	1	1	1			0.5	2.5	Animation (0.5p). Simple ASCII PLY loader (2.0p)
918875	10	10	0	1	3	3	3		R1: translation only in 1D (-0p)								
930484	11	10	1	1	3	3	3		VCS: Would have been nicer to give a logfile, instead of having the TA view the logs themselves on a terminal...	1							
932440	18	10	8	1	3	3	3		Rotate&Scale: can't find a way to rotate. Your instructions only rotate the camera. Camera is quite simple. Perspective: no fov-slider.	1	0.5		1.5	2	1	2	Animation (0.5p). Simple ASCII PLY loader (1.5p).
935625	11	10	1	1	3	3	3			1							
939375	0	0	0	0	0	0	0		Submission had no source files :(								

Student number	point total	req total	extra total	R1 moving (1p)	R2 cone (3p)	R3 unpack (3p)	R4 loader (3p)	mod	notes	VCS (1p)	Rotate and scale (1p)	Normal trans. in shader (1p)	Camera (max 3p)	Normal trans. in uniform (2p)	Viewport & perspective (0.5-2p)	other (put points here)	what other extras?
942618	14	10	4	1	3	3	3		Normal transform: normal not normalized afterwards.	1	1			1.5		0.5	Animation (0.5p)
k28342	0	0	0														