Student no.	point total	req total	extra total	R1 Texture sampling (1p)	R2 diffuse shading (1p)	R3 D (1p)	R4 G (1p)	R5 Fr (1p)	mod	notes	World space (2p, 3p if optimized)	Point lights (3p)	Moving lights (1p)	SSS (2p)	Color/position variations (1.5 p)	Shadow maps (4p)	Shadowmap SSS (2p)	Envmap (3+p)	Other extras (? p)	What other extras
225157	0	0	0																	
270034	2	2	0	1	1	0	0	0												
293846	0	0	0																	
295323	0	0	0																	
345642	0	0	0																	
348843	0	0	0																	
349936	0	0	0																	
350475	5	5	0	1	1	1	1	1												
352091	0	0	0																	
353980	0	0	0																	
354439	0	0	0																	
355593	0	0	0																	
356026	0	0	0																	
361749	0	0	0																	
369181	5	5	0	1	1	1	1	1												
372660	0	0	0																	
387370	0	0	0																	
425575	0	0	0																	
425614	0	0	0																	
426419	0	0	0																	
427489	0	0	0																	
428022	0	0	0																	
429487	0	0	0																	
430829	5	5	0	1	1	1	1	1												
457598	0	0	0																	
460297	0	0	0																	
464772	0	0	0																	
46477D	0	0	0																	
46596K	0	0	0																	
474199	0	0	0																	
474322	0	0	0																	
474458	0	0	0																	
474898	0	0	0																	
475389	0	0	0																	
475813	0	0	0																	
475910	0	0	0																	
477329	6.5	5	1.5	1	1	1	1	1		Color variation: very erratic, makes little sense visually (only applies to strip on backside of head)			1		0.5					
477811	0	0	0																	
478328	0	0	0																	
478470	0	0	0																	
478687	0	0	0																	
479505	12.5	5	7.5	1	1	1	1	1		Smile looks cool!. Shadow maps: VS: need to left-multiply with posToLightClip, uv also incorrectly scaled		3	1		1.5	2				
479589	0	0	0																	
479741	0	0	0																	
480086	0	0	0																	
480248	0	0	0																	
480714	4.5	4.5	0	0.5	1	1	1	1		R1: read texture normal never assigned to mappedNormal (which is where a second transform is needed). The existing code ("line 117") transforms the interpolated per-vertex normal										
480798	0	0	0																	

Student no.	point total	req total	extra total	R1 Texture sampling (1p)	R2 diffuse shading (1p)	R3 D (1p)	R4 G (1p)	R5 Fr (1p)	mod	notes	World space (2p, 3p if optimized)	Point lights (3p)	Moving lights (1p)	SSS (2p)	Color/position variations (1.5 p)	Shadow maps (4p)	Shadowmap SSS (2p)	Envmap (3+p)	Other extras (? p)	What other extras
481577	0	0	0																	
493840	4	4.5	0	1	1	1	1	0.5	-0.5	Specular term does not contain lambertian cosine (-0.5p). R5: Fr does not match reference, seems entirely black (logic looks OK though)										
506300	4.5	4.5	0	1	0.5	1	1	1		R3: Your formula looks quite different from the one in the handout (where did the exponential function e^x come from?)										
508285	0	0	0																	
51620U	0	0	0																	
524926	0	0	0																	
525417	2	2	0	1	1															
525491	0	0	0																	
525941	0	0	0																	
526050	4.5	5	0	1	1	1	1	1	-0.5	Specular never added to end result, also missing lambertian cosine term (-0.5)										
526319	0	0	0																	
526775	0	0	0																	
527143	0	0	0																	
527389	0	0	0																	
528867	0	0	0																	
528883	0	0	0																	
529196	0	0	0																	
529303	0	0	0																	
530185	0	0	0																	
530363	0	0	0																	
530619	0	0	0																	
530648	0	0	0																	
530868	5	5	0	1	1	1	1	1												
530981	0	0	0																	
540094	0	0	0																	
540654	0	0	0																	
544566	0.5	0.5	0	0.5						R1: almost; on line 128, you need to replicate the logic of line 117.										
549040	0	0	0																	
549163	0	0	0																	
55055P	0	0	0																	
552794	6.5	5	1.5	1	1	1	1	1		World space: need to pass camera position to shaders to compute V. Currently impossible to know if there are other bugs since the shading is black	1.5									
552969	0	0	0																	
554598	0	0	0																	
563068	0	0	0																	
576149	0	0	0																	
585716	0	0	0																	
586333	0	0	0																	
586702	0	0	0																	
586980	4.5	5	0	1	1	1	1	1	-0.5	Specular term never used, missing lambertian cosine (-0.5p)										
587316	0	0	0																	
587471	5	5	0	1	1	1	1	1												

Student no.	point total	req total	extra total	R1 Texture sampling (1p)	R2 diffuse shading (1p)	R3 D (1p)	R4 G (1p)	R5 Fr (1p)	mod	notes	World space (2p, 3p if optimized)	Point lights (3p)	Moving lights (1p)	SSS (2p)	Color/position variations (1.5 p)	Shadow maps (4p)	Shadowmap SSS (2p)	Envmap (3+p)	Other extras (? p)	What other extras
588289	4.5	5	0	1	1	1	1	1	-0.5	Specular term never used, missing lambertian cosine (-0.5p)										
589291	0	0	0																	
589343	0	0	0																	
589848	0	0	0																	
590921	0	0	0																	
591904	2.5	2.5	0	1	1	0.5				R3: you need acos to get theta (or you can use one of the mentioned identities)										
591946	0	0	0																	
592929	0	0	0																	
593274	0	0	0																	
593847	0	0	0																	
594435	0	0	0																	
595926	5	5	0	1	1	1	1	1												
595997	0	0	0																	
596747	0	0	0																	
597429	0	0	0																	
597623	0	0	0																	
597937	9	5	4	1	1	1	1	1								4				
602851	0	0	0																	
602893	0	0	0																	
603096	7.5	5	2.5	1	1	1	1	1					1		1.5					
603245	13.5	4.5	9.5	0.5	1	1	1	1	-0.5	R1: texture normal needs to be left-multiplied with normalToCamera (order matters). Specular term missing lambertian cosine (-0.5p). World space: code looks OK, but does not match camera-space lighting exactly (they should be identical)	2.5	3	1	2					1	Colors YT video (1p)
604095	15	5	10	1	1	1	1	1		World space: V seems to be a vector pointing towards the origin in world space - you need to take the camera position into account as well.. SSS: not properly calibrated, effect is way too strong. Shadows: logic seems mostly OK, but lightUV calculation is wrong (division by w and scaling to [0,1] missing)	2.5	3	1	1		2.5				
604273	4	4	0	1	1	0.5	1	0.5		R3: H vector incorrectly normalized (should be normalize(L+V)), V points in wrong direction. Fresnel: integer division (1/2) causes term to become zero										
606064	5	5	0	1	1	1	1	1		Thanks for your feedback, you make valid points. Perhaps a glossier material would make the effect of the specular term more obvious and rewarding to mimplement										
608949	0	0	0																	
609155	0	0	0																	
609249	0	0	0																	
612472	0	0	0																	
612498	0	0	0																	
612870	5	5	0	1	1	1	1	1												
614577	0	0	0																	

Student no.	point total	req total	extra total	R1 Texture sampling (1p)	R2 diffuse shading (1p)	R3 D (1p)	R4 G (1p)	R5 Fr (1p)	mod	notes	World space (2p, 3p if optimized)	Point lights (3p)	Moving lights (1p)	SSS (2p)	Color/position variations (1.5 p)	Shadow maps (4p)	Shadowmap SSS (2p)	Envmap (3+p)	Other extras (? p)	What other extras
614580	0	0	0																	
621308	5	5	0	1	1	1	1	1												
628835	5	5	0	1	1	1	1	1												
63036R	0	0	0																	
641922	0	0	0																	
646804	0	0	0																	
647764	0	0	0																	
648530	0	0	0																	
648860	0	0	0																	
650191	0	0	0																	
650227	0	0	0																	
650405	0	0	0																	
650560	5	5	0	1	1	1	1	1												
650942	11.5	4.5	7	1	1	0.5	1	1		R3: positionVarying is already in camera coords, should not transform again. Matrix multiplication is not in general commutative, don't mix Tx and xT! Shadowmaps: The way you construct the view matrix is also somewhat involved: it is just m_xform. inverted()! Could not locate the bug on a quick look. Otherwise code looks reasonable.		3	1			3				
651527	0	0	0																	
651585	8	5	3	1	1	1	1	1		Shadows: getPosToLightClip not implemented, looks OK otherwise						3				
651637	4	4	0	1	1	1	1	0		As per your question, yes. However, only for the final rendering. For R3/4 visualizations we just want the the contributions of G and D.										
651789	0	0	0																	
652102	2	2	0	1	1	0	0	0												
652131	4.5	4.5	0	0.5	1	1	1	1		R1: matrix multiplication is not in general commutative, should use Tx instead of xT. R3: PositionVarying is already in camera space, should not transform again! Round 1 extra: unfortunately the way you construct the matrices is incorrect. Also, for composition of transforms you need to multiply the transforms, not sum :(										
652335	0	0	0																	
652649	0	0	0																	
652898	0	0	0																	
652937	4.5	5	0	1	1	1	1	1	-0.5	Specular term never used, missing lambertian cosine (-0.5p)										
653127	8	5	3	1	1	1	1	1			3									
653596	0	0	0																	
653693	8	4	4	0.5	1	0.5	1	1		R1: you are not trasforming the normals to camera space. R3: your formula looks fishy and the visual results don't agree. Specular term is never actually added to final rendering.				2					2	RK4 (2p)

Student no.	point total	req total	extra total	R1 Texture sampling (1p)	R2 diffuse shading (1p)	R3 D (1p)	R4 G (1p)	R5 Fr (1p)	mod	notes	World space (2p, 3p if optimized)	Point lights (3p)	Moving lights (1p)	SSS (2p)	Color/position variations (1.5 p)	Shadow maps (4p)	Shadowmap SSS (2p)	Envmap (3+p)	Other extras (? p)	What other extras
653871	0	0	0																	
653907	0	0	0																	
653910	0	0	0																	
654595	17.5	5	12.5	1	1	1	1	1		Shadow maps: should not use abs() in pixel shader. getPosToLightClip is also broken (missing at least one inverse() of the xform)	3	3	1	2	1.5	2				
655057	0	0	0																	
655086	0	0	0																	
655109	2	2	0	1	1															
655251	0	0	0																	
655264	0	0	0																	
655471	0	0	0																	
655691	0	0	0																	
655853	0	0	0																	
656250	0	0	0																	
656454	15.5	5	10.5	1	1	1	1	1		Point lights: you are computing vecToLight between a normalized light_direction_ and positionVarying (position), which does not really make sense. Shadowmaps: you need not touch the z-values of your shadow map in vshader. Only map x and y to [0,1].	3	1.5	1	2		3				
656616	8	5	3	1	1	1	1	1				3								
657291	0	0	0																	
657314	0	0	0																	
657327	5	5	0	1	1	1	1	1												
657482	0	0	0																	
657796	4	4	0	0.5	1	0.5	1	1		R1+3: matrix multiplication is not in general commutative, should use Tx instead of xT. R3: PositionVarying is already in camera space, should not transform again!									
657893	0	0	0																	
659914	4.5	4.5	0	1	1	1	0.5	1		R4: don't quite undestand using min instead if product in G. Also potentially a bit inefficient to use trig and inverse-trig functions when you can avoid them with trig-identities (from the handout)									
660246	0	0	0																	
660877	0	0	0																	
660893	0	0	0																	
663191	0	0	0																	
663272	0	0	0																	
665380	0	0	0																	
665898	2	2	0	1	1	0	0	0												
666172	0	0	0																	
666350	7	5	2	1	1	1	1	1						2						
666680	4.5	4.5	0	1	1	1	1	0.5		R5: specular term not multiplied by the lambertian cosine.										
667249	0	0	0																	
67137M	0	0	0																	
67627H	5	5	0	1	1	1	1	1												
677734	0	0	0																	

Student no.	point total	req total	extra total	R1 Texture sampling (1p)	R2 diffuse shading (1p)	R3 D (1p)	R4 G (1p)	R5 Fr (1p)	mod	notes	World space (2p, 3p if optimized)	Point lights (3p)	Moving lights (1p)	SSS (2p)	Color/position variations (1.5 p)	Shadow maps (4p)	Shadowmap SSS (2p)	Envmap (3+p)	Other extras (? p)	What other extras
678089	0	0	0																	
68933B	0	0	0																	
69247N	0	0	0																	
700436	0	0	0																	
705570	34.5	5	29.5	1	1	1	1	1		Great work all around! As for your question: it depends. If you are more into research, then probably a mathematics or algorithms track (albeit not directly related, they are essential tools!). If you are into games, then perhaps look into the game design and production program. Come to the advanced class and consider doing a special assignment on graphics!	3	3	1	2	1.5	4	2		13	Shadowmap filtering for smoother shadows (1p), Tangent space normal maps (3p), Texture-space diffusion SSS (5p), Deferred shading (4p)
706566	0	0	0																	
708784	11.5	4.5	7	1	0.5	1	1	1		R2: the light direction calculation looks very dangerous. It goes right if normalize(lightPositions[i]) = lightDirection[i] (for directional lights), which is not in general the case. Here, it looks like the directional lights happen to have positions where this works. Point lights: distance computation uses lightTypes[i]?	3	3	1							
708904	0	0	0																	
708920	0	0	0																	
708933	0	0	0																	
709291	0	0	0																	
709628	0	0	0																	
710086	4.5	4.5	0	0.5	1	1	1	1		R1: normals in world space instead of camera space.										
710497	4.5	4.5	0	1	1	0.5	1	1		R3: posVarying is already in camera coords from vertex shader - should not transform it again.										
710743	0	0	0																	
710976	0	0	0																	
711182	0	0	0																	
711467	12.5	4.5	8	1	1	0.5	1	1		R3: posVarying is already in camera coords, you should not try to transform it again. World space lighting: you are doing both the camera and normal space computations, so not too optimized! You could also just bring the camera location from the CPU to avoid taking the posToWorld*[posVayrying, 1] -product in the shader.	2	3	1	2						
711551	0	0	0																	
711810	0	0	0	0	0	0	0	0		Code seems to be unmodified base code?										
711904	0	0	0																	
712550	0	0	0																	
712686	0	0	0																	
712819	5.5	4.5	1	1	1	0.5	1	1		R3-5: positionVarying is already in camera space from vertex shader, should not transform it again.			1							
712958	0	0	0																	

Student no.	point total	req total	extra total	R1 Texture sampling (1p)	R2 diffuse shading (1p)	R3 D (1p)	R4 G (1p)	R5 Fr (1p)	mod	notes	World space (2p, 3p if optimized)	Point lights (3p)	Moving lights (1p)	SSS (2p)	Color/position variations (1.5 p)	Shadow maps (4p)	Shadowmap SSS (2p)	Envmap (3+p)	Other extras (? p)	What other extras
713672	0	0	0																	
714985	5	5	0	1	1	1	1	1												
716080	0	0	0																	
716462	4.5	4.5	0	1	1	0.5	1	1		R3: posVarying is already in camera coords from vertex shader - should not transform it again.										
716718	0	0	0																	
716860	0	0	0																	
717377	0	0	0																	
717474	0	0	0																	
717513	5	5	0	1	1	1	1	1												
717539	0	0	0																	
718020	0	0	0																	
718871	0	0	0																	
722427	0	0	0																	
723691	19.5	5	14.5	1	1	1	1	1		World-space: could maybe just give the camera location as uniform to avoid doing cameraToWorld*[0,0,0,1] in shader.	3	3	1	2	1.5	4				
723905	0	0	0																	
728667	0	0	0																	
728900	0	0	0																	
729132	1	2	0	1	1	0	0	0	-1	Missing .sln-files.										
729967	0	0	0																	
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	0	0	0																	
76509T	5	5	0	1	1	1	1	1												
765510	5	5	0	1	1	1	1	1												
766331	0	0	0																	
767042	0	0	0																	
767136	2	2	0	1	1					R2: your code is correct, you just need to add the difuse lighting into 'light_contribution'										
768504	0	0	0																	
769396	0	0	0																	
77388B	7	4.5	2.5	0.5	1	1	1	1		R1: texture normal needs to be left-multiplied with matrix (order matters). RK4: mixing f's and x's together, they have separate roles (derivatives vs states, i.e. directions vs positions in state space)			1						1.5	Round 1: animation (0.5 p), rot and scale (1p). Round 4: RK4 (+0p)
779124	0	0	0																	
780058	0	0	0																	
780346	14	5	9	1	1	1	1	1								4	2		3	Assn5: indirect illumination (3p)
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																	

Student no.	point total	req total	extra total	R1 Texture sampling (1p)	R2 diffuse shading (1p)	R3 D (1p)	R4 G (1p)	R5 Fr (1p)	mod	notes	World space (2p, 3p if optimized)	Point lights (3p)	Moving lights (1p)	SSS (2p)	Color/position variations (1.5 p)	Shadow maps (4p)	Shadowmap SSS (2p)	Envmap (3+p)	Other extras (? p)	What other extras
788380	0	0	0																	
788678	0	0	0																	
791982	0	0	0																	
795700	0	0	0																	
795755	0	0	0																	
796039	0	0	0																	
804183	5	5	0	1	1	1	1	1												
829155	0	0	0																	
838191	0	0	0																	
83873J	0	0	0																	
84308F	0	0	0																	
84858E	0	0	0																	
848754	4.5	4.5	0	0.5	1	1	1	1		R1: normal texture not used.										
875170	5	5	0	1	1	1	1	1		R3-5: you only needed to visualize specular*CookTorrance without without the lambertian cosine and light contribution. Apart from that, all is good.										
875251	5	5	0	1	1	1	1	1		Could just call normalize to get unit vectors. No need to compute lengths elementwise by hand!										
875303	0	0	0																	
875617	0	0	0																	
876399	0	0	0																	
877107	0	0	0																	
877152	0	0	0																	
878591	5	5	0	1	1	1	1	1												
878627	0	0	0																	
878889	0	0	0																	
879105	0	0	0																	
882134	0	0	0																	
885128	5	5	0	1	1	1	1	1												
886648	0	0	0																	
889645	0	0	0																	
892292	0	0	0																	
898351	0	0	0																	
899130	0	0	0																	
900016	12.5	5	7.5	1	1	1	1	1		World space lighting: a lot of weirdness going on - you are multiplying V from the right with normalToCamera. Since world to camera is only rotation + translation, the inverse transpose of the upper 3x3 part is just the rotation component of worldToCamera. Now, since you are multiplying from the right, this is the equivalent of multiplying by worldToCamera.T, which is actually the inverse - and you end up with cameraToWorld*direction_in_camera_space. So after all these lucky cancellations you end up doing the right thing. With a more general transformation this would not work and I have hard time believing this was intentional.	1.5	3	1	2						

Student no.	point total	req total	extra total	R1 Texture sampling (1p)	R2 diffuse shading (1p)	R3 D (1p)	R4 G (1p)	R5 Fr (1p)	mod	notes	World space (2p, 3p if optimized)	Point lights (3p)	Moving lights (1p)	SSS (2p)	Color/position variations (1.5 p)	Shadow maps (4p)	Shadowmap SSS (2p)	Envmap (3+p)	Other extras (? p)	What other extras
901170	0	0	0																	
901196	0	0	0																	
913249	0	0	0																	
913333	0	0	0																	
913346	0	0	0																	
913566	5	5	0	1	1	1	1	1												
915221	0	0	0																	
915250	0	0	0																	
917863	0	0	0																	
918150	0	0	0																	
918228	4.5	4.5	0	1	1	1	1	0.5		Specular term not multiplied by the lambertian cosine and not added to the final rendering.										
918257	0	0	0																	
918309	30	5	25	1	1	1	1	1		World space lighting: no, nothing should change! (think about why that is). Point lights: it is quite confusing to have a variable pointLightDirections that actually carries positions (and is used as such)...	3	3	1	2	1.5				14.5	RK4 (2p), Wind (2p), Spray system (3p), Fog (1p), Refraction (2p), More primitives (3p), Transparent shadows (1.5p)
918396	0	0	0																	
918464	0	0	0																	
918671	0	0	0																	
918875	10.5	5	5.5	1	1	1	1	1		World space lighting: positionVarying is still in camera coordinates so V is wrong, otherwise looks ok.	1.5	3	1							
930484	0	0	0																	
932440	4	4	0	0.5	1	0.5	1	1		R1: multiplying normalTexture with normalToCamera from the right. Matrix multiplication is not in general commutative! R3: positionVarying is already in camera space, should not transform again!										
935625	0	0	0																	
939375	0	0	0																	
942618	2	2	0	1	1	0	0	0												
k28342	0	0	0																	