1. В Domain-шейдере

2. Получим корректные границы объектов с выпуклостями

3. Он быстрее рендерится

4. Из окружения (удаляем наш объект и ставим вместо него камеру, в которую рендерим окружающие текстуры для отражения): с 6 сторон (справа-слева, сверху-снизу, назад-вперед)

5. Когда идет рендеринг во все стороны, полученная текстура на самом деле не лежит на гранях куба, поэтому выходят проекции, просто картинки, не учитывающие расстояния до объектов. Поэтому близкие объекты (~радиус объекта) искажаются и становятся в проекции такими же, как и далекие.

6. В пиксельном шейдере в текстуре берем окрестность точки, в которую попали отраженным лучом (тексели в окрестности точки с некоторым радиусом), осредняем их и получаем нужный цвет.

7. Вода прозрачная => из окружения (комбинируем преломленный цвет неба, цвет дна) и отраженный цвет (например, луч, попавший на гору)

8. Вычисляем высоты по пространственным (x, y z) и текстурным (u, v) координатам (берем синус Kx\*t). K – скорость распространения волны (берем разные). В каждый момент t берем в каждой точке высоту, взятую из синуса. Получим синусную волну, затем найдем суперпозицию по разным скоростям волн. Затем берем комбинации синусов, двигающихся в разные стороны с подобранными фазами, и получаем уже готовые волны.

9. По среднему значению освещенности сцены

10. Ставим камеру на источник света, рендерим объект на источник света – получаем текстуру. На текстуре Pixel-шейдер помечает 1 – если пиксель в тени, 0 – если не в тени. Интерпретируем текстуру как clamp – если попали мимо, берем цвет с границы.