








Implementation of an Individual English Oral Training Robot System

Chen-Yu Lin¹ , Wei-Wei Shen² , Ming-Hsiu Michelle Tsai² ,
Jim-Min Lin¹ , and Wai Khuen Cheng³ 

¹ Department of Information Engineering and Computer Science, Feng Chia University, Taichung City 40724, Taiwan

qwel11845@gmail.com, jimmy@fcu.edu.tw

² Department of Foreign Languages and Literature, Feng Chia University, Taichung City 40724, Taiwan

{wwshen, mh-tsai}@fcu.edu.tw

³ Faculty of Information and Communication Technology, Universiti Tunku Abdul Rahman, Kampar, Malaysia

chengwk@utar.edu.my

Abstract. To improve oral English ability, in addition to learners' willingness to practice more, the learning effect will be more obvious if tutor assistance is provided and can involve one-on-one individual tutoring. However, due to the scarcity of English teacher manpower, teachers cannot take care of every student in class, nor can they teach students one by one after class. Robot-Assisted English Speaking may provide a feasible solution. Therefore, this research has developed an educational robot system that can support individual tutoring of English speaking after class. It is called "English Oral Training Robot Tutor System (EOTRTS)", which can actively lead students to learn through robots, and help students improve their oral English practice through individual tutoring and interactive methods. The implementation of this system is to use the social robot NAO's voice recognition, QR code scanning, humanized limbs, and various sensor functions as well as the ability to interact with people and other features to help students learn oral English after class. The experimental results show that, in addition to a slightly lower satisfaction with the robot's gesture performance, the students' acceptance of the EOTRTS system is promising.

Keywords: Educational robot · Robot assisted language learning · English Oral Training Robot Tutor System (EOTRTS) · Google Cloud Automatic Speech Recognizer (ASR) · NAO

1 Introduction

With the development of technology, many robots have been used in real life today, and the existence of social robots allows us to interact with them through dialogue or touch. The interaction with the robot has a positive impact on the body and mind [1]. Mubin et al. [2] also enumerate the various advantages of using robots in a teaching environment, such as dull and boring tasks without fatigue and remote teaching. Kang et al. [3] assumes that the robot is humanized and has the proper appearance to interact

directly with the person. Extending these ideas, robots are an attractive tool that is brought into the second language learning arena and used to meet different teaching needs. Kanda et al. [4] and others placed robots in the primary school classroom for two weeks and compared the frequency of interactions between students and their English test scores. Although the two-week robot-assisted learning did not have any significant impact on students' oral English and listening test scores, students who showed great interest at the initial stage had significantly improved English scores.

In the previous research, most of the robots were used in the classroom. The purpose of using robots in the classroom was to attract students' attention and enhance the interest of learning. An ETAR system [4] was implemented to be a teaching assistant in class. It can help to improve the motivation of English learning in class and bring convenience to classroom teaching for English teachers. Research [5] was also proved to be effective. However, the robot cannot take care of every student in the classroom; likewise, after the class, the teacher cannot teach each student individually. Therefore, a robot system for after-school tutoring, called English Oral Training Robot Tutor System (EOTRTS) [6] is proposed. This paper presents how the system is implemented. In this system, the educational robot leads the students to learn in a one-to-one manner, using a social robot NAO with interactive functions such as voice function and touch as a personal lecturer for the learner to conduct after-school tutoring for oral English training. Different from the past research, the robot instructor can actively guide the learner to learn. It does not need to be triggered by the helper to guide the coaching process. Instead, the robot instructor NAO actively guides the students according to the progress of the student's curriculum to practice single words, recitation of texts, recording exercises, and evaluate the student's speech rate and correct reading rate through Google Cloud Voice to Text and Word Error Rate Calculation Formula.

The structure of this article is as follows. Section 2 explains how the system is designed, and describes the design of the student's interaction with the robot. The details of the experimental procedures is described in Sect. 3. Next, the analysis and discussion of the result are then explained in Sect. 4. Finally, a brief conclusion is made and the future research direction is explored in Sect. 5.

2 System Design

2.1 System Structure

Figure 1 shows the system structure of the educational robot (called EOTRTS) proposed in this study. This article uses NAO robots to conduct after-school tutoring and lead students to learn text materials. EOTRTS is mainly divided into three parts: cloud services, servers, and robots. The system software functions are implemented in Python language, and Socket is used as a network communication mechanism to connect various system components. Cloud services use Google Cloud services.

Server. The server is responsible for accessing the database and the cloud, obtaining course data to send to the robot, recording the session test data returned by the robot, etc., and respectively storing each student's recording files after reading the text. After

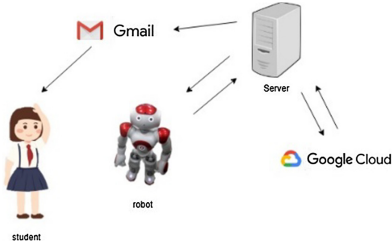


Fig. 1. System structure of EOTRTS.

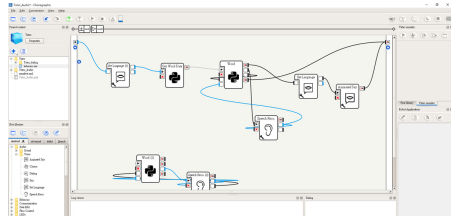


Fig. 2. Choregraphe operational interface.

the server uploads the data and the data reaches the cloud, it uses Google Cloud Speech to Text (STT) and Word Error Rate (WER) formulas to calculate the accuracy of reading aloud and speaking rate of the text recording, and sends the recognition results of the text and the recognition results of the conversation test through SMTP (Simple Mail Transfer Protocol) after the session test of the unit is over.

Robot. The robot end mainly designs the functional process of the robot, including scanning the QR code to log in to the account, requesting the server to confirm the account, and returning the course progress and course data. The TTS technology of the Nao robot itself is used for the pronunciation of the words, and the storage of audio files, playback, and recording are implemented using the NAOqi SDK. Google Cloud STT is used for the real-time recognition function of the conversation test. After the result is recognized, the recording file is read to calculate the reading speed, and the result is finally returned to the server.

- Hardware – NAO robot

The NAO robot has enough sensors, so it is quite helpful for this research. This research uses sensors such as head and hand to enhance the interaction of tactile sense. It is also possible to use touch to continue the course to enhance the motivation of students to learn. There are 6 touch points (Head-front, Head-middle, Head-rear, Left hand, Right-hand, Right foot) provided for users to interact with EOTRTS.

- Software development – Choregraphe

EOTRTS uses the Choregraphe development platform because Choregraphe can directly use local calls to create modules for the use of various methods. The program in Choregraphe is coded in a way that each functional module has a block, and then it controls the process by connecting various modules. Therefore, it is relatively easy to know how each function is performed in terms of the program structure, and the process of after-school tutoring can also be simply modified according to the needs of the tutor. Choregraphe can visualize each function using the Python programming language. Figure 2 shows the operational interface of Choregraphe platform.

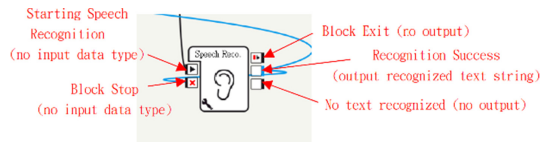


Fig. 3. Speech recognition block in choregraphe.

```

1  # -*- coding: utf-8 -*-
2  from threading import Lock
3  import sys
4  import os
5  import re
6  import time
7  import subprocess
8  import sys
9  import os
10 import re
11 import time
12 import subprocess
13 import sys
14 import os
15 import re
16 import time
17 import subprocess
18 import sys
19 import os
20 import re
21 import time
22 import subprocess
23 import sys
24 import os
25 import re
26 import time
27 import subprocess
28 import sys
29 import os
30 import re
31 import time
32 import subprocess
33 import sys
34 import os
35 import re
36 import time
37 import subprocess
38 import sys
39 import os
40 import re
41 import time
42 import subprocess
43 import sys
44 import os
45 import re
46 import time
47 import subprocess
48 import sys
49 import os
50 import re
51 import time
52 import subprocess
53 import sys
54 import os
55 import re
56 import time
57 import subprocess
58 import sys
59 import os
60 import re
61 import time
62 import subprocess
63 import sys
64 import os
65 import re
66 import time
67 import subprocess
68 import sys
69 import os
70 import re
71 import time
72 import subprocess
73 import sys
74 import os
75 import re
76 import time
77 import subprocess
78 import sys
79 import os
80 import re
81 import time
82 import subprocess
83 import sys
84 import os
85 import re
86 import time
87 import subprocess
88 import sys
89 import os
90 import re
91 import time
92 import subprocess
93 import sys
94 import os
95 import re
96 import time
97 import subprocess
98 import sys
99 import os
100 import re
101 import time
102 import subprocess
103 import sys
104 import os
105 import re
106 import time
107 import subprocess
108 import sys
109 import os
110 import re
111 import time
112 import subprocess
113 import sys
114 import os
115 import re
116 import time
117 import subprocess
118 import sys
119 import os
120 import re
121 import time
122 import subprocess
123 import sys
124 import os
125 import re
126 import time
127 import subprocess
128 import sys
129 import os
130 import re
131 import time
132 import subprocess
133 import sys
134 import os
135 import re
136 import time
137 import subprocess
138 import sys
139 import os
140 import re
141 import time
142 import subprocess
143 import sys
144 import os
145 import re
146 import time
147 import subprocess
148 import sys
149 import os
150 import re
151 import time
152 import subprocess
153 import sys
154 import os
155 import re
156 import time
157 import subprocess
158 import sys
159 import os
160 import re
161 import time
162 import subprocess
163 import sys
164 import os
165 import re
166 import time
167 import subprocess
168 import sys
169 import os
170 import re
171 import time
172 import subprocess
173 import sys
174 import os
175 import re
176 import time
177 import subprocess
178 import sys
179 import os
180 import re
181 import time
182 import subprocess
183 import sys
184 import os
185 import re
186 import time
187 import subprocess
188 import sys
189 import os
190 import re
191 import time
192 import subprocess
193 import sys
194 import os
195 import re
196 import time
197 import subprocess
198 import sys
199 import os
200 import re
201 import time
202 import subprocess
203 import sys
204 import os
205 import re
206 import time
207 import subprocess
208 import sys
209 import os
210 import re
211 import time
212 import subprocess
213 import sys
214 import os
215 import re
216 import time
217 import subprocess
218 import sys
219 import os
220 import re
221 import time
222 import subprocess
223 import sys
224 import os
225 import re
226 import time
227 import subprocess
228 import sys
229 import os
230 import re
231 import time
232 import subprocess
233 import sys
234 import os
235 import re
236 import time
237 import subprocess
238 import sys
239 import os
240 import re
241 import time
242 import subprocess
243 import sys
244 import os
245 import re
246 import time
247 import subprocess
248 import sys
249 import os
250 import re
251 import time
252 import subprocess
253 import sys
254 import os
255 import re
256 import time
257 import subprocess
258 import sys
259 import os
260 import re
261 import time
262 import subprocess
263 import sys
264 import os
265 import re
266 import time
267 import subprocess
268 import sys
269 import os
270 import re
271 import time
272 import subprocess
273 import sys
274 import os
275 import re
276 import time
277 import subprocess
278 import sys
279 import os
280 import re
281 import time
282 import subprocess
283 import sys
284 import os
285 import re
286 import time
287 import subprocess
288 import sys
289 import os
290 import re
291 import time
292 import subprocess
293 import sys
294 import os
295 import re
296 import time
297 import subprocess
298 import sys
299 import os
300 import re
301 import time
302 import subprocess
303 import sys
304 import os
305 import re
306 import time
307 import subprocess
308 import sys
309 import os
310 import re
311 import time
312 import subprocess
313 import sys
314 import os
315 import re
316 import time
317 import subprocess
318 import sys
319 import os
320 import re
321 import time
322 import subprocess
323 import sys
324 import os
325 import re
326 import time
327 import subprocess
328 import sys
329 import os
330 import re
331 import time
332 import subprocess
333 import sys
334 import os
335 import re
336 import time
337 import subprocess
338 import sys
339 import os
340 import re
341 import time
342 import subprocess
343 import sys
344 import os
345 import re
346 import time
347 import subprocess
348 import sys
349 import os
350 import re
351 import time
352 import subprocess
353 import sys
354 import os
355 import re
356 import time
357 import subprocess
358 import sys
359 import os
360 import re
361 import time
362 import subprocess
363 import sys
364 import os
365 import re
366 import time
367 import subprocess
368 import sys
369 import os
370 import re
371 import time
372 import subprocess
373 import sys
374 import os
375 import re
376 import time
377 import subprocess
378 import sys
379 import os
380 import re
381 import time
382 import subprocess
383 import sys
384 import os
385 import re
386 import time
387 import subprocess
388 import sys
389 import os
390 import re
391 import time
392 import subprocess
393 import sys
394 import os
395 import re
396 import time
397 import subprocess
398 import sys
399 import os
400 import re
401 import time
402 import subprocess
403 import sys
404 import os
405 import re
406 import time
407 import subprocess
408 import sys
409 import os
410 import re
411 import time
412 import subprocess
413 import sys
414 import os
415 import re
416 import time
417 import subprocess
418 import sys
419 import os
420 import re
421 import time
422 import subprocess
423 import sys
424 import os
425 import re
426 import time
427 import subprocess
428 import sys
429 import os
430 import re
431 import time
432 import subprocess
433 import sys
434 import os
435 import re
436 import time
437 import subprocess
438 import sys
439 import os
440 import re
441 import time
442 import subprocess
443 import sys
444 import os
445 import re
446 import time
447 import subprocess
448 import sys
449 import os
450 import re
451 import time
452 import subprocess
453 import sys
454 import os
455 import re
456 import time
457 import subprocess
458 import sys
459 import os
460 import re
461 import time
462 import subprocess
463 import sys
464 import os
465 import re
466 import time
467 import subprocess
468 import sys
469 import os
470 import re
471 import time
472 import subprocess
473 import sys
474 import os
475 import re
476 import time
477 import subprocess
478 import sys
479 import os
480 import re
481 import time
482 import subprocess
483 import sys
484 import os
485 import re
486 import time
487 import subprocess
488 import sys
489 import os
490 import re
491 import time
492 import subprocess
493 import sys
494 import os
495 import re
496 import time
497 import subprocess
498 import sys
499 import os
500 import re
501 import time
502 import subprocess
503 import sys
504 import os
505 import re
506 import time
507 import subprocess
508 import sys
509 import os
510 import re
511 import time
512 import subprocess
513 import sys
514 import os
515 import re
516 import time
517 import subprocess
518 import sys
519 import os
520 import re
521 import time
522 import subprocess
523 import sys
524 import os
525 import re
526 import time
527 import subprocess
528 import sys
529 import os
530 import re
531 import time
532 import subprocess
533 import sys
534 import os
535 import re
536 import time
537 import subprocess
538 import sys
539 import os
540 import re
541 import time
542 import subprocess
543 import sys
544 import os
545 import re
546 import time
547 import subprocess
548 import sys
549 import os
550 import re
551 import time
552 import subprocess
553 import sys
554 import os
555 import re
556 import time
557 import subprocess
558 import sys
559 import os
560 import re
561 import time
562 import subprocess
563 import sys
564 import os
565 import re
566 import time
567 import subprocess
568 import sys
569 import os
570 import re
571 import time
572 import subprocess
573 import sys
574 import os
575 import re
576 import time
577 import subprocess
578 import sys
579 import os
580 import re
581 import time
582 import subprocess
583 import sys
584 import os
585 import re
586 import time
587 import subprocess
588 import sys
589 import os
590 import re
591 import time
592 import subprocess
593 import sys
594 import os
595 import re
596 import time
597 import subprocess
598 import sys
599 import os
600 import re
601 import time
602 import subprocess
603 import sys
604 import os
605 import re
606 import time
607 import subprocess
608 import sys
609 import os
610 import re
611 import time
612 import subprocess
613 import sys
614 import os
615 import re
616 import time
617 import subprocess
618 import sys
619 import os
620 import re
621 import time
622 import subprocess
623 import sys
624 import os
625 import re
626 import time
627 import subprocess
628 import sys
629 import os
630 import re
631 import time
632 import subprocess
633 import sys
634 import os
635 import re
636 import time
637 import subprocess
638 import sys
639 import os
640 import re
641 import time
642 import subprocess
643 import sys
644 import os
645 import re
646 import time
647 import subprocess
648 import sys
649 import os
650 import re
651 import time
652 import subprocess
653 import sys
654 import os
655 import re
656 import time
657 import subprocess
658 import sys
659 import os
660 import re
661 import time
662 import subprocess
663 import sys
664 import os
665 import re
666 import time
667 import subprocess
668 import sys
669 import os
670 import re
671 import time
672 import subprocess
673 import sys
674 import os
675 import re
676 import time
677 import subprocess
678 import sys
679 import os
680 import re
681 import time
682 import subprocess
683 import sys
684 import os
685 import re
686 import time
687 import subprocess
688 import sys
689 import os
690 import re
691 import time
692 import subprocess
693 import sys
694 import os
695 import re
696 import time
697 import subprocess
698 import sys
699 import os
700 import re
701 import time
702 import subprocess
703 import sys
704 import os
705 import re
706 import time
707 import subprocess
708 import sys
709 import os
710 import re
711 import time
712 import subprocess
713 import sys
714 import os
715 import re
716 import time
717 import subprocess
718 import sys
719 import os
720 import re
721 import time
722 import subprocess
723 import sys
724 import os
725 import re
726 import time
727 import subprocess
728 import sys
729 import os
730 import re
731 import time
732 import subprocess
733 import sys
734 import os
735 import re
736 import time
737 import subprocess
738 import sys
739 import os
740 import re
741 import time
742 import subprocess
743 import sys
744 import os
745 import re
746 import time
747 import subprocess
748 import sys
749 import os
750 import re
751 import time
752 import subprocess
753 import sys
754 import os
755 import re
756 import time
757 import subprocess
758 import sys
759 import os
760 import re
761 import time
762 import subprocess
763 import sys
764 import os
765 import re
766 import time
767 import subprocess
768 import sys
769 import os
770 import re
771 import time
772 import subprocess
773 import sys
774 import os
775 import re
776 import time
777 import subprocess
778 import sys
779 import os
780 import re
781 import time
782 import subprocess
783 import sys
784 import os
785 import re
786 import time
787 import subprocess
788 import sys
789 import os
790 import re
791 import time
792 import subprocess
793 import sys
794 import os
795 import re
796 import time
797 import subprocess
798 import sys
799 import os
800 import re
801 import time
802 import subprocess
803 import sys
804 import os
805 import re
806 import time
807 import subprocess
808 import sys
809 import os
810 import re
811 import time
812 import subprocess
813 import sys
814 import os
815 import re
816 import time
817 import subprocess
818 import sys
819 import os
820 import re
821 import time
822 import subprocess
823 import sys
824 import os
825 import re
826 import time
827 import subprocess
828 import sys
829 import os
830 import re
831 import time
832 import subprocess
833 import sys
834 import os
835 import re
836 import time
837 import subprocess
838 import sys
839 import os
840 import re
841 import time
842 import subprocess
843 import sys
844 import os
845 import re
846 import time
847 import subprocess
848 import sys
849 import os
850 import re
851 import time
852 import subprocess
853 import sys
854 import os
855 import re
856 import time
857 import subprocess
858 import sys
859 import os
860 import re
861 import time
862 import subprocess
863 import sys
864 import os
865 import re
866 import time
867 import subprocess
868 import sys
869 import os
870 import re
871 import time
872 import subprocess
873 import sys
874 import os
875 import re
876 import time
877 import subprocess
878 import sys
879 import os
880 import re
881 import time
882 import subprocess
883 import sys
884 import os
885 import re
886 import time
887 import subprocess
888 import sys
889 import os
890 import re
891 import time
892 import subprocess
893 import sys
894 import os
895 import re
896 import time
897 import subprocess
898 import sys
899 import os
900 import re
901 import time
902 import subprocess
903 import sys
904 import os
905 import re
906 import time
907 import subprocess
908 import sys
909 import os
910 import re
911 import time
912 import subprocess
913 import sys
914 import os
915 import re
916 import time
917 import subprocess
918 import sys
919 import os
920 import re
921 import time
922 import subprocess
923 import sys
924 import os
925 import re
926 import time
927 import subprocess
928 import sys
929 import os
930 import re
931 import time
932 import subprocess
933 import sys
934 import os
935 import re
936 import time
937 import subprocess
938 import sys
939 import os
940 import re
941 import time
942 import subprocess
943 import sys
944 import os
945 import re
946 import time
947 import subprocess
948 import sys
949 import os
950 import re
951 import time
952 import subprocess
953 import sys
954 import os
955 import re
956 import time
957 import subprocess
958 import sys
959 import os
960 import re
961 import time
962 import subprocess
963 import sys
964 import os
965 import re
966 import time
967 import subprocess
968 import sys
969 import os
970 import re
971 import time
972 import subprocess
973 import sys
974 import os
975 import re
976 import time
977 import subprocess
978 import sys
979 import os
980 import re
981 import time
982 import subprocess
983 import sys
984 import os
985 import re
986 import time
987 import subprocess
988 import sys
989 import os
990 import re
991 import time
992 import subprocess
993 import sys
994 import os
995 import re
996 import time
997 import subprocess
998 import sys
999 import os
1000 import re

```

Fig. 4. Corresponding program python codes of Fig. 3

As shown in Fig. 2, there are many boxes, each of which is a module with the specified functions. By connecting each box to integrate the system, the box is normally the code, but it can also be Block flow chart as in the middle of Fig. 3. The advantage of developing on Choregraphe is that you can see which module is currently actively running, which is convenient for programmers to control the process of the system. However, the design of each module is not as simple as in writing usual programs. It is necessary to set the input data type and output format with a specific code-writing method for each module. Figure 3 and Fig. 4 illustrate the functional design examples of the block. Figure 3 is the speech recognition block, and Fig. 4 is the compiled version of the corresponding Python codes.

The entry point and exit point of each recognition box have their own input and output to be set, which have different colors according to different output data types. For example, the output string is blue; the type and length of the output data needs to be properly set in advance, otherwise the program will have an error occurred or fail to receive the data that should be obtained. For example, the voice recognition entry point in Fig. 4 will go to `def onInput_onStart(self)`: The method in this line will start to operate. If one want to get data, he must set the data input in the box and set the data type of the team, and then change the Python code part to `def onInput_onStart(self, data)`: to get the transferred data.

The above is briefly the preliminary process of designing each block. There are requirements to use the microphone and so on. The *ALSpeechRecognition* and the *ALMemory* modules can be used through the above mentioned NAOqi methods. Then the voice recognition function can be turned on through the methods inside.

Cloud Services. An important function of this system—“voice recognition”—is provided through cloud services. The robot saves the recording file of each student on the server after the text is read aloud, and then uploads it to the cloud service from the server. We use Google Cloud Speech-to-Text (STT) service to recognize the text recording and then convert it into a corresponding text file. Finally, we use the Word Error Rate (WER) formula to calculate the accuracy and speed of the student’s reading aloud. After the end of the session test of a unit, the recognition result of the text read aloud and the session test will be sent to the student via SMTP (Simple Mail Transfer Protocol) by email.

- Google Cloud Platform Speech-To-Text (STT) Service

Google Cloud Platform has a forward-looking infrastructure, powerful data analysis services. EOTRTS uses Google Cloud Automatic Speech Recognizer (ASR) for speech recognition and its STT API for speech to text translation, in which a powerful neural network is added to the Cloud STT API. This model is convenient for developers to convert their audio messages into text. Additionally, because it is a voice recognition model, to have a higher recognition rate, it requires a complete and clear pronunciation for single words to be read. Therefore, it is very suitable for students to practice reading aloud. As a result, students would be able to read English clearly and completely, thereby improving their reading ability. This API mainly has two functions, namely:

- Synchronous voice recognition: For short voice message (less than 1 min), the recognized text will be recognized by a built-in STT function in NAO, and returned in the response immediately. EOTRTS uses this part in the robot conversation test. This function is particularly suitable for the conversation test as a conversation has fewer sentences and needs to be quickly identified and then sent back.
- Asynchronous voice recognition: Performing voice recognition on batches of long audio files will start long-running audio processing operations. Asynchronous voice recognition is used to recognize an audio message that is longer than 1 min and stored in Google Cloud Storage. The system then uses this API to identify the part of the text to be practiced in the system.

The recognition rate can be obtained by comparing the original text with its recording of student’s read aloud voice. The ASR recognition rate is more than 80%, even up to as high as 92%, which is an acceptable recognition rate.

When using asynchronous voice recognition, only the audio files stored in Google Cloud Storage. The system creates a *bucket* named *speech_to_text_class* on the cloud platform to store the student audio files to be processed. Then we can upload audio files to this bucket from the server. In order for the data to be reusable, we use the *Regional* level storage space and grant permissions through Cloud Storage ID, so that project members can access Cloud Storage data based on their project roles. Server-end account access rights are also granted through Cloud Storage ID.

- Correct rate of reading aloud and calculation of speaking speed

WER is a common measurement of speech recognition or machine translation system performance. It was proposed by Levenshtein [7]. The calculation formula is as Eq. (1). The calculation method is as follows:

S is the number of different words in the original sentence, D is the number of missing words, I is the number of inserted words, and N is the number of correct words. The spaces, commas, and other symbols are removed in the calculation of correct rate.

$$WER = 100 * \frac{S + D + I}{N} \% \quad (1)$$

The formula for calculating the speed of reading aloud is as Eq. (2), and Eq. (3).

$$Speaking\ Speed = \frac{text\ length\ (words)}{recording\ time(min)},\ if\ \geq\ 1\ minute \quad (2)$$

$$Speaking\ Speed = 60 * \frac{text\ length(words)}{recording\ time(min)},\ if\ <\ 1\ minute \quad (3)$$

2.2 English Oral Educational Functions in EOTRST

EOTRST has 4 main oral English education functions: single-word pronunciation exercises, text reading exercises, text recordings, and conversation tests. The robots in the process have operation tips to prevent students from forgetting how to operate them.

- Single-word pronunciation exercises

After entering the course practice, students begin to learn the vocabulary in the new course. The robot reads the words first, and the students follow along. Students learn the meaning of the words, so that the students will not have difficulty to read the new words in the text reading practice afterwards. After reading a word, there is a waiting interval of 3 s for students to practice the pronunciation of the word, and then NAO will read the next word. Vocabulary practice is about two and a half minutes, and students can follow their own abilities to choose whether to practice again.

- Text read-aloud exercises

The student touches NAO's head to enter the text reading exercise. The NAO robot will play the text audio files (real-person English pronunciation) of the practice unit, and can make some actions to attract the students' attention. It will stop at the appropriate semantic paragraph. After students listen to the audio file and practice reading the passage, they can listen to the audio file of the next passage by touching NAO's right hand or right foot. If they don't understand this passage, they can repeat this audio file by touching NAO's head (front). Then NAO will play the text audio file of the whole unit. If students want to practice again, they can touch NAO's head (middle) to listen to the entire audio file again. The estimated practice time for this process is around 10 min.

- Text read-aloud recordings

When the text reading exercise is over, the NAO robot prompts, “Touch my front head to start recording.” After that, NAO will not take any action and wait. When a student thinks that he/she can start the next process, he/she will touch NAO’s front head to start recording. After recording, the student can touch NAO’s middle head to stop recording. Students choose whether to re-record the read-aloud again orally by saying “YES” or “NO”. If students do not choose to re-record, the robot will upload the audio recording file to the server.

- Conversation tests

EOTRTS will firstly read the questions of the conversation test in the textbook. In order to train the student’s listening ability, the test paper we give will only have the answer part, and the question will be read by the robot. The robot reads the question twice with 2 s apart each time. Student will first answer which choice (A-D) it is, and the robot will ask the student to read the complete sentence of that answer choice next. For example, if the student says, “A”, the robot will ask the student to read the sentence of his answer choice. Students touch NAO’s forehead to activate its short speech recognition function, and touch the middle of NAO’s head to end the function after finishing reading. After that, an email with the text recognition content of the unit and the recognition content of the conversation test will be sent immediately to the student.

3 EOTRTS Experiment

3.1 Participants

The experiment is aimed at students in the science-related departments of a university in central Taiwan. The total number of students is 19, including 17 boys and 2 girls, with an average age of 21.65 years and a standard deviation of 1.06. A total of 19 people in the experimental group and the control group participated in the experiment.

3.2 Teaching Material

The textbook used in this study is 4000 Essential English Words 1. It’s an English textbook selected for freshmen. It has a certain correlation with the participating students. Students have certain familiarity with the teaching materials, and the difficulty of the teaching materials is relatively low. It is suitable for students of science and engineering background for the purpose of improving their insufficient oral English ability.

3.3 Procedure

The experiment lasted a total of 6 weeks. In the first week, the overall experimental process and the pre-test group were introduced and the English learning motivation and English learning anxiety questionnaire were filled out. In the 2nd to 5th weeks, the experimental group and the control group were arranged in a quiet and undisturbed

room space. The experimental group used EOTRTS for learning, while the control group learned English in a self-study manner. Participants spent half an hour per unit, and 1 h for two units per week. There were four main learning steps for each unit: single-word pronunciation practice, text reading practice, text reading-aloud practice and conversation test. In addition, the two groups of students recorded their text reading-aloud practices, and the session tests were recorded afterwards as well. The experimental group used the robot to record, and the control group used the mobile phone to record. In the sixth week, the two groups of students engaged in post-testing, and filled out the English learning motivation and English learning anxiety questionnaires, while the experimental group also filled out the technology acceptance questionnaire, and the control group filled out the learning satisfaction questionnaire.

4 Results: Analysis of Technology Acceptance Questionnaire

Limited by the paper length, this paper will only report the analysis of the Technology Acceptance Questionnaire. More results will be presented in our future papers. This questionnaire is designed based on the five-point Likert scale. All questionnaires have been evaluated by two English teachers to ensure that the questions are clear and structured. The technology acceptance questionnaire was conducted in the experimental group to understand students' satisfaction and acceptance of EOTRTS. The average of the item classification is shown in Table 1. Table 2 shows the questions with answers of lower mean.

Table 1. Technology acceptance questionnaire topic classification.

Evaluation items	Ease of use	Usefulness
Sound	4.3	4.2
Action	3.4	3.9
Procedure	4.4	4.3
Interaction	4.1	4.1

In terms of ease of use, except for the part of the gesture “action”, the rest of the evaluation items receive all 4.0 or more, which means that the usability in sound, process, interaction, results and other items is acceptable. However, the action items got the lowest point at 3.4. After asking most students in the experimental group, the reason is that they think the robot had too many actions and big moves in its gestures. As shown in Table 2, it is indeed as low as 3.6 and 3.3 points.

In terms of usefulness, except for the part of gesture action, the other evaluation items all receive points above 4.0, which means that the usefulness of items such as sound, procedure, interaction, and results also has a good effect. Although the robot's action can attract the attention of students, its excessive movements can also distract students. The lowest-rated item in the questionnaire is the STT ability to recognize speech. This is partly because EOTRTS is recognized through the STT function of

Table 2. Technology acceptance questionnaire with answers of lower mean.

No	Questions	1	2	3	4	5	Mean
3	I think the speed of the Educational robot is appropriate	11.1%	11.1%	22.2%	22.2%	33.3%	3.6
4	I think that the display of Educational robot movements with spoken content is appropriate	11.1%	11.1%	22.2%	44.4%	11.1%	3.3
14	I am satisfied with the system ability to identify the voice recognition function of the server (the identification result sent back by email)	0%	22.2%	33.3%	11.1%	33.3%	3.6
15	I am satisfied with the Educational robot's ability to read English in real time	0%	22.2%	44.4%	22.2%	11.1%	3.2

strongly disagree (1) → strongly agree (5)

Google Cloud. It would have a great impact on the recognition of short speech, including the speech sound level, environmental noise, and even the name of the person, etc. Because the conversation test answer is a sentence, up to a dozen words, as long as one or two words are not recognized, it would have a very big impact on the scoring. Therefore, it would also affect the real-time STT recognition greatly, as shown in Table 2 in which the average score is only 3.2. On the contrary, the impact of that on the server's text reading-aloud recording recognition ability is relatively lowered. Because each text is more than 300 words, the impact on names or small number of errors is not very large, and the average recognition rate is 0.4 higher than that of real-time recognition.

5 Conclusion

This article proposes an educational robot system EOTRTS to provide individual tutoring to students to improve their oral English ability. It also proposes to use Google Cloud-STT and WER calculation formulas to evaluate the accuracy of students' reading aloud, so as to improve the accuracy of their English reading and reading rate. After six weeks of experiments, EOTRTS can indeed play the role of a tutor, helping students learn English in a one-on-one manner. The results of questionnaires show that, regardless of ease-of-use and usefulness, this system has good system acceptance for users, except for the robot action factor. It signifies that using robots as a language learning tool is a new way of oral English learning that can be accepted by students.

Future Works. EOTRTS can incorporate more robots with different functions, appearances, and operating types in the future to support language learning on different robots. In addition, EOTRTS can also be used for different foreign languages in the future, such as Japanese and Korean, etc. because NAO robots can support multiple languages. Furthermore, in the learning process, many people are attracted by the vivid and interesting appearance and actions of the robot and the lively and delicate nature of the interaction between the robot and the human. Therefore, EOTRTS can also be combined with the previous research results of [8–10] in the future to design more lively and vivid robot programs.

Acknowledgement. This study is partly supported in finance by the Ministry of Science and Technology, TAIWAN under the contract numbers MOST106-2511-S-035-003-MY2, MOST107-2511-H-035-004-MY2, and MOST108-2511-H-035-002-MY2.

References

1. Grewen, K.M., Anderson, B.J., Girdler, S.S., Light, K.C.: Warm partner contact is related to lower cardiovascular reactivity. *Behav. Med.* **29**(3), 123–130 (2003)
2. Mubin, O., Shahid, S., Bartneck, C.: Robot assisted language learning through games: a comparison of two case studies. *Aust. J. Intell. Inform. Process. Syst.* **13**, 9–14 (2013)
3. Kang, S.C., Chang, W.T., Gu, K.Y., Chi, H.L.: *Robot Development Using Microsoft Robotics Developer Studio*. CRC Press, Boca Raton (2016)
4. Kanda, T., Hirano, T., Eaton, D., Ishiguro, H.: Interactive robots as social partners and peer tutors for children: a field trial. *Hum.-Comput. Interact.* **19**, 61–84 (2004)
5. Shen, W.W., Tsai, M.M., Wei, G.C., Lin, C.Y., Lin, J.M.: ETAR: an english assistant robot and its effects on college freshmen's in-class learning motivation. In: L. Rønningsbakk et al. (eds.): *ICITL 2019, LNCS 11937*, pp. 77–86 (2019)
6. Lin, C.Y.: *EOTRTS: Research into an English Oral Training Robot Tutor System to Support After-Class Individual Learning*. Master Thesis, Feng Chia University, Taiwan (2019)
7. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. *Sov. Phys. Dokl.* **10**(8), 707–710 (1966)
8. Li, Y.H., Lin, J.M., Lee, K.Y., Chiou, C.W.: A robot motion control mechanism using robot drama markup language. *J. Comput. Appl. Sci. Educ.* **5**(2), 17–35 (2018)
9. Luo, B., Shen, W.W.: Repeated reading: a practical approach when using an ESL textbook? *Feng Chia J. Hum. Soc. Sci.* **30**, 77–103 (2015)
10. Wei, G.C., Shen, W.W., Tsai, M.H.M., Lin, J.M.: An education robot supporting adaptive learning for english as a foreign language. In: Lin, J.M., Chang, C.-C., Hwang, Y.M. (eds.) *The 9th Conference on Engineering, Technological and Technology Education CETTE 2020*, pp. 44–55 (2020)